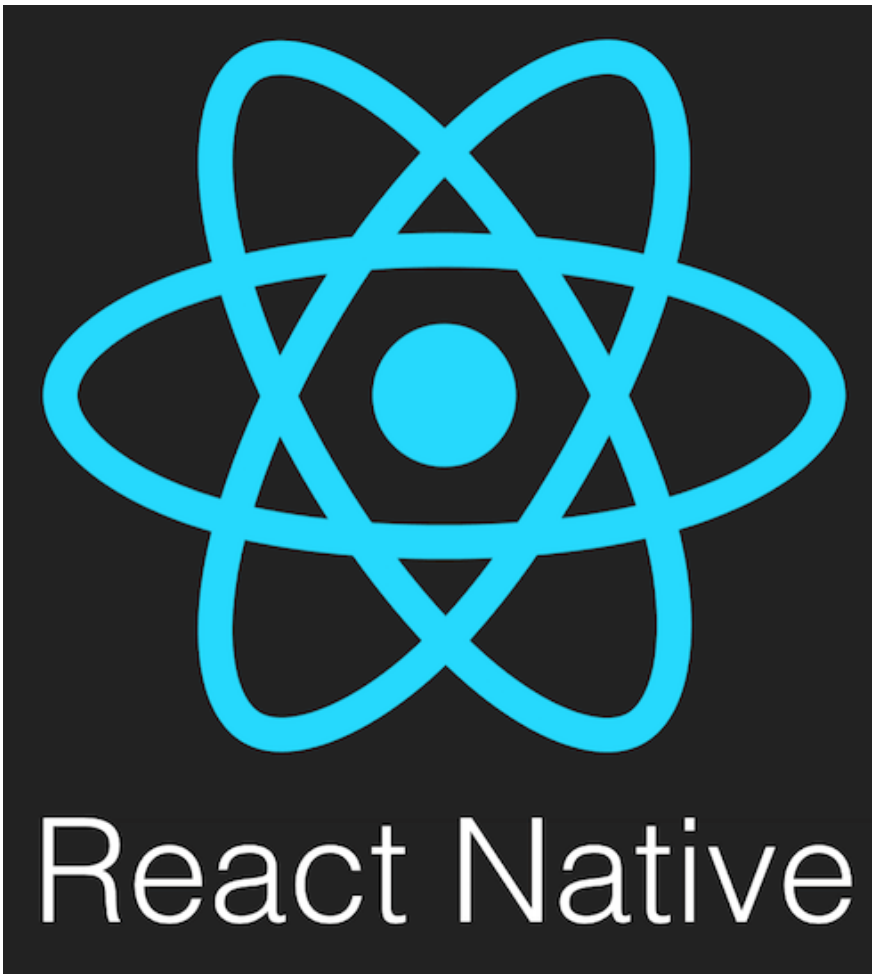
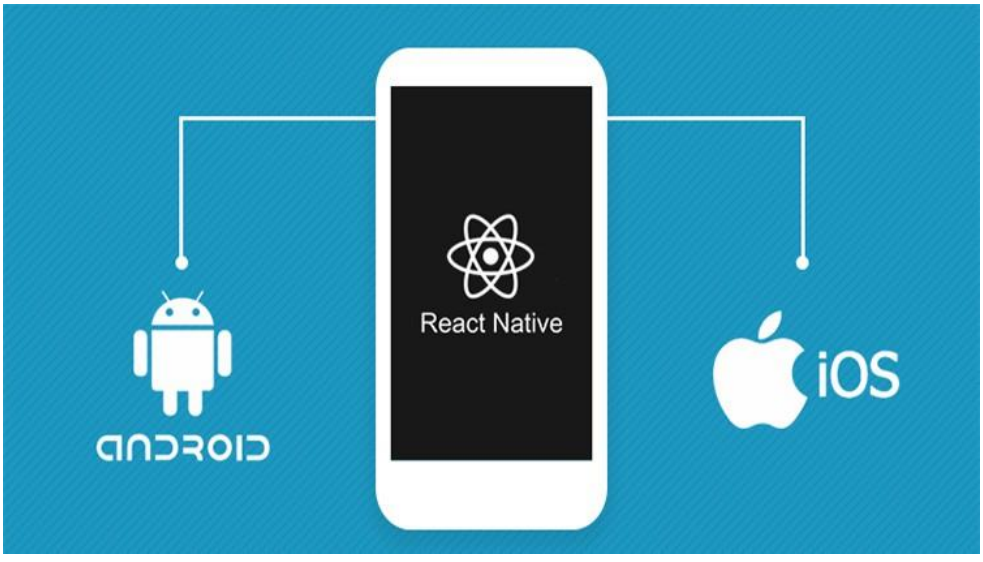
Desenvolvimento Mobile



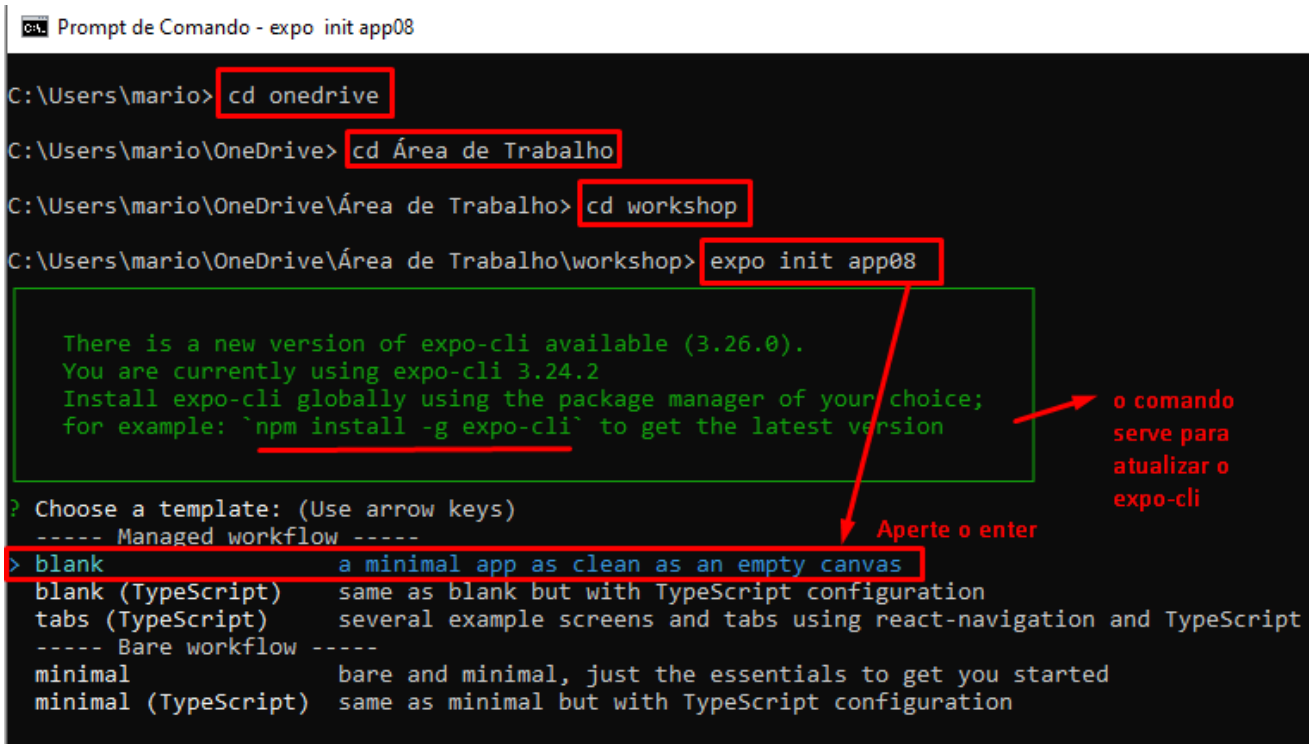


Professor: Mário de Jesus

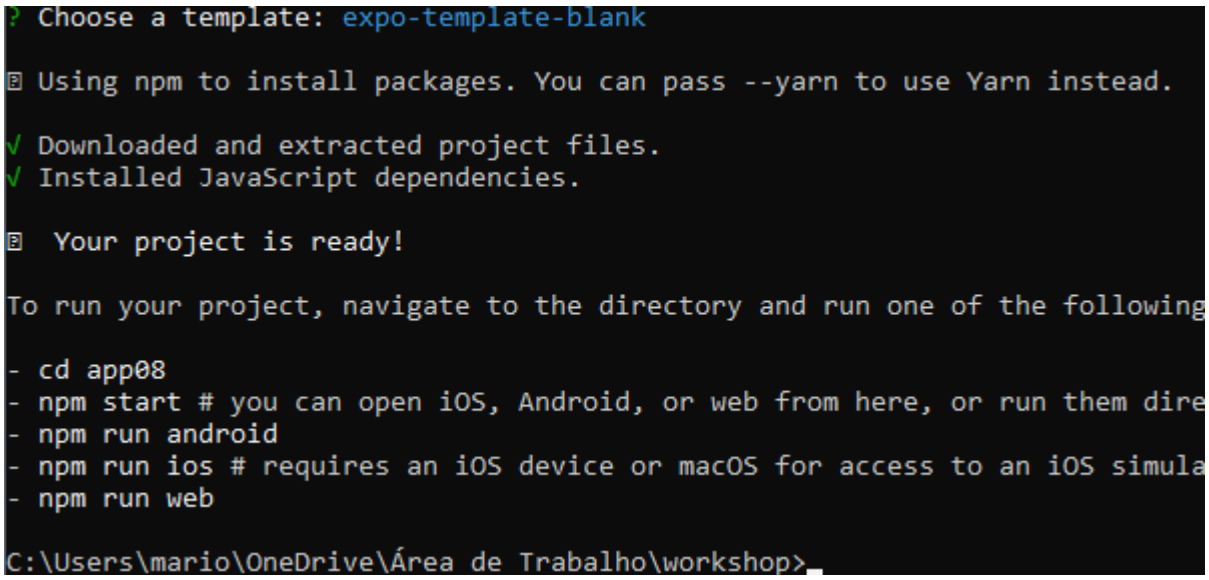
Professor: Caio Malheiros

Vamos criar o app08 como trabalhar com classes

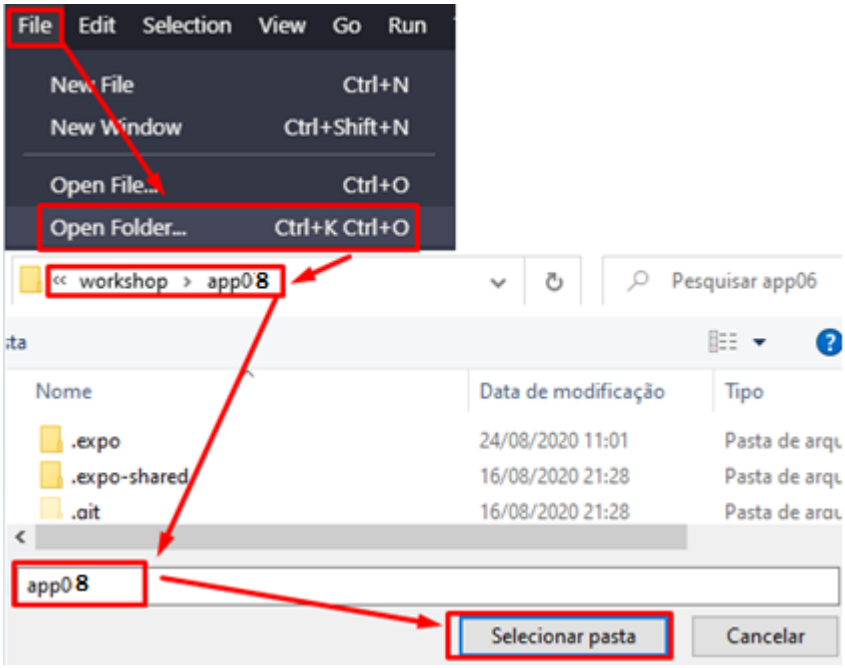
Agora vamos criar nosso aplicativo nessa pasta usando o comando **expo init "nome do projeto"**



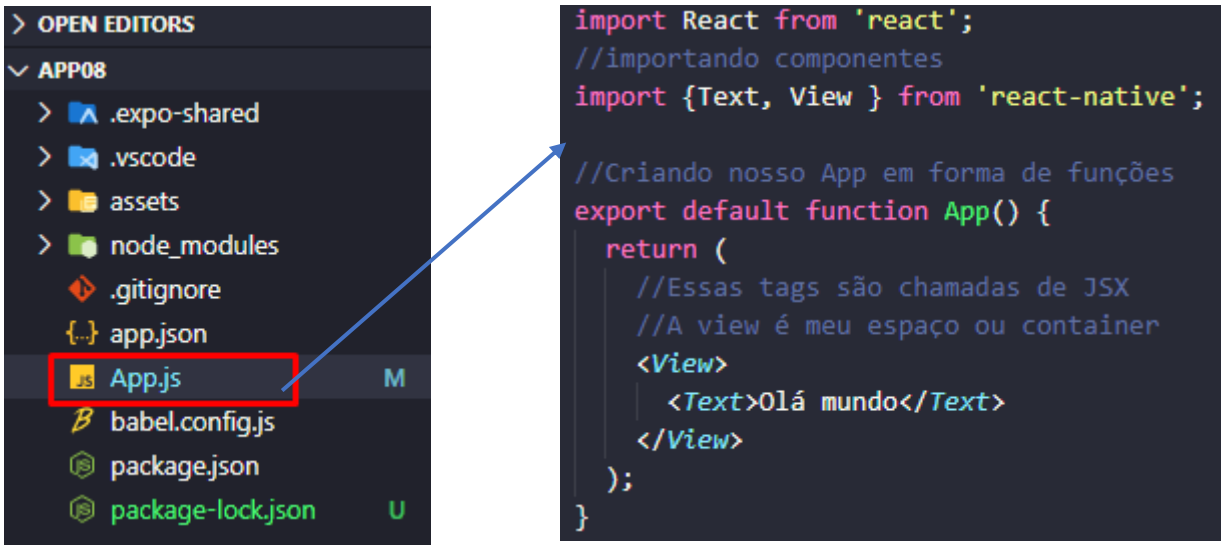
Aperte **enter** para ficar assim:



Abra o **visual studio code** e depois abra a pasta **app08** dentro de workshop

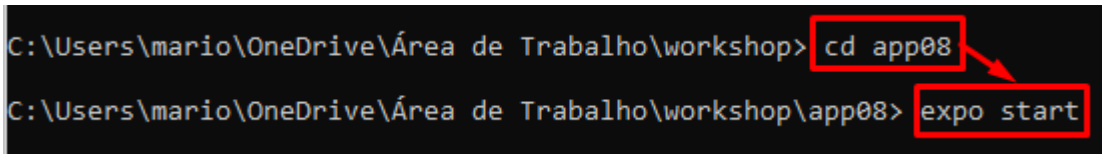


Abra o App.js – e digite



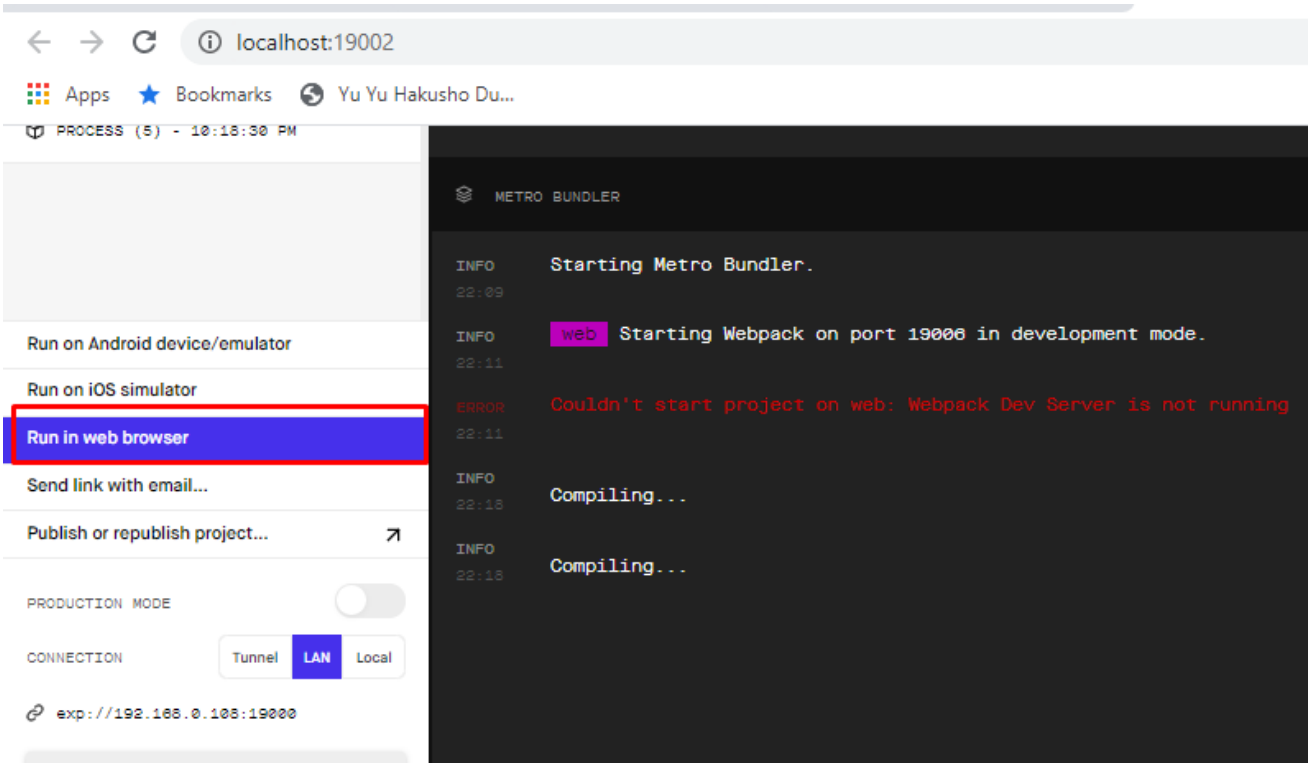
Importante: Perceba que até agora nós **importamos os componentes**, **programamos dentro de uma função** chamada **App** e digitamos nosso **código** usando **tags JSX**.

Agora vamos **entrar** na pasta **app08** e **rodar** nosso **projeto** pelo **CMD**

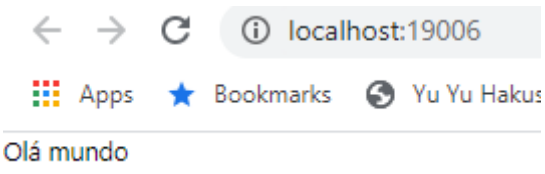


Executando pelo browser

Agora vamos rodar na Web, depois quem quiser pode passar para o celular usando o QRCode



Veja o resultado



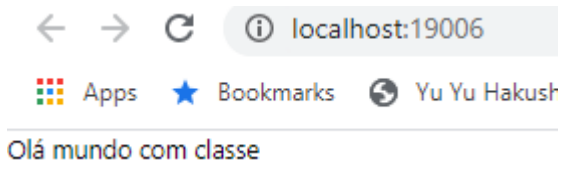
Trabalhando com Classes usando Component

Agora vamos ter **componentes** em **forma** de **classe**. Para isso precisamos usar o **Component**.

Altere o **App.js** novamente e digite o código abaixo:

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

//Criando nosso App em forma de Classe
export default class App extends Component {
  render(){
  return (
    //Essas tags são chamadas de JSX
    //A view é meu espaço ou container
    <View>
      <Text>Olá mundo com classe</Text>
    </View>
  );
  }
}
```

Veja o resultado no browser



Trabalhando com imagens externas

Agora vamos trabalhar com imagens externas. Altere novamente o App.js

```
import React, {Component} from 'react';
//importando componentes
import {Text, View, Image } from 'react-native';

//Criando nosso App em forma de Classe
export default class App extends Component {
  render(){
    let nome = 'Profº Mario';
    let img = 'https://sujeitoprogramador.com/steve.png'
    return (
      <View>
        <Text>Meu App</Text>
        <Text style={{ color: '#FF0000', fontSize: 25, margin: 15}}>Programador</Text>

        <Image
          source={{ uri: img}}
          style={{ width: 300, height: 300}}
        />
        <Text style={{ fontSize: 30}}> {nome} </Text>

      </View>
    );
  }
}
```

Salve, e veja o resultado no browser

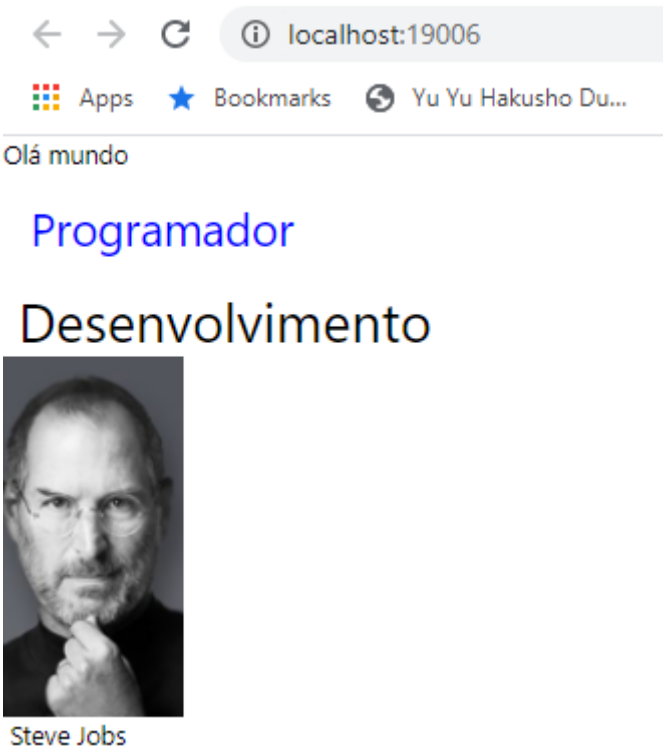
Trabalhando com props

Os **props** são propriedades onde nós podemos passar ou utilizar informações. As **props** são estáticas, ou seja, se eu receber um valor **externo** e quiser alterá-lo isso não será possível.

Altere o App.js novamente

```
import React, {Component} from 'react';
//importando componentes
import {Text, View, Image } from 'react-native';

//Criando nosso App em forma de Classe
export default class App extends Component {
  render(){
    let nome = 'Desenvolvimento';
    return (
      <View>
        <Text>Olá mundo</Text>
     <Text style={{ color: '#0000ff' ,fontSize: 25, margin: 15 }}>Programador</Text>

        <Text style={{ fontSize: 30}}> {nome} </Text>

        <Jobs largura={100} altura={200} fulano="Steve Jobss" />

      </View>
    );
  }
}

class Jobs extends Component{
  render(){
    let img = 'https://sujeitoprogramador.com/steve.png';
    return(
      <View>
        <Image
          source={{ uri: img}}
          style={{ width: this.props.largura, height: this.props.altura}}
        />
        <Text> {this.props.fulano}</Text>
      </View>
    );
  }
}
```

Salve e veja o resultado no browser



Trabalhando com states

Como vimos as **props** são estáticas, ou seja, não podem ser alteradas por ações externas. Os **states** são estados e ao contrário das props eles são **mutáveis**, ou seja, podemos trocar o valor dessas variáveis utilizando ações externas.

Altere o App.js novamente

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

//Criando nosso App em forma de Classe
export default class App extends Component {

  constructor(props){
    super(props);
    this.state={
      nome: 'Mário de Jesus'
    };
    //fazendo isso a função vai acessar
    //todas as propriedades
    this.entrar = this.entrar.bind(this);
  }
  entrar(){
    this.setState({
      nome: 'Caio Malheiros'
    })
  }
  render(){

    return (
      <View style={{ marginTop: 20}}>

        <Button title="Entrar" onPress={this.entrar}/>

        <Text style={{ color: 'red' ,fontSize: 25, textAlign: 'center' }}>
          {this.state.nome}
        </Text>

      </View>
    );
  }
}
```

Salve e veja o resultado no browser

Trabalhando com states usando parâmetros

Vamos fazer algumas alterações para passar o argumento em forma de parâmetro usando uma função anomia em forma de arrow function. Usando o **this.entrar('Programando App')**

Altere o App.js novamente

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

//Criando nosso App em forma de Classe
export default class App extends Component {

  constructor(props){
    super(props);
    this.state={
      nome: 'Mário de Jesus'
    };
    //fazendo isso a função vai acessar
    //todas as propriedades
    this.entrar = this.entrar.bind(this);
  }
  entrar(nome){
    this.setState({
      nome: nome
    });
  }
  render(){

    return (
      <View style={{ marginTop: 20}}>

        <Button title="Entrar" onPress={ () => this.entrar('Programando App')}/>

        <Text style={{ color: 'red' ,fontSize: 25, textAlign: 'center' }}>
          {this.state.nome}
        </Text>

      </View>
    );
  }
}
```

Salve e veja o resultado no browser