# Paradigmas de Programação
# React Native

## Axios: App Contatos – 2ª Parte

*Utilizando Satck Navigator*

**Gil Eduardo de Andrade**

# APP: CONTATOS – STACKNAVIGATOR

**Descrição:**

- O ***Stack Navigator*** (*createStackNavigator*) disponibiliza um sistema simples de controle para transição de telas dentro da aplicação;
- Ele utiliza uma estrutura de pilha para armazenar a sequência de telas visualizada, e assim identificar qual tela deve ser invocada quando usuário seleciona a opção voltar;

# APP: CONTATOS – STACKNAVIGATOR

## Instalação:

- A instalação do componente ***Stack Navigator*** pode ser feita via comando ***npm***:

  – ***npm install --save react-navigation***;

# createStackNavigator na Prática

## Contatos: App

www.gileduardo.com.br/ifpr/pp_rn/downloads/pp_rn_exapp10.zip

# STACK NAVIGATOR: CONFIGURAÇÃO

## Definição das Telas *(App.js)*

```
import ListaContatos from './src/screens/ListaContatos';
import Contato from './src/screens/Contato';

export default createStackNavigator(
  {
    'Main' : {
        screen : ListaContatos
    },
    'Contato' : {
      screen : Contato,
      navigationOptions : ( {navigation} ) => {
        const nome = UpperCase(navigation.state.params.pessoa.name.first);

        return ({
          title : nome
        });
      }
    }
  },
```

No arquivo App.js definimos (configuramos) as telas que farão parte da aplicação, nesse caso *"Main"* e *"Contato"*. Por questões de organização de código, as duas telas foram criadas e colocadas na pastas *"/src/screens"*. A tela nomeada como *"Main"* faz referência ao arquivo *"ListaContatos.js"*, enquanto a tela nomeada como *"Contato"* faz referência ao arquivo *"Contato.js"*.

Por padrão, o *Navigator* invoca, para exibição inicial da aplicação, a primeira tela especificada em sua configuração, no caso do exemplo a tela *"Main"*. Ao especificarmos uma tela é possível efetuarmos alguma configurações também, via propriedade *"navigationOptions"*. Observe que no código acima estamos definindo o *título* (nome do contato) que será exibido pela tela do barra superior do *Navigator*. Isso é possível porque ele permite a passagem de dados entre telas via *{navigation.state}*.

# STACK NAVIGATOR: CONFIGURAÇÃO

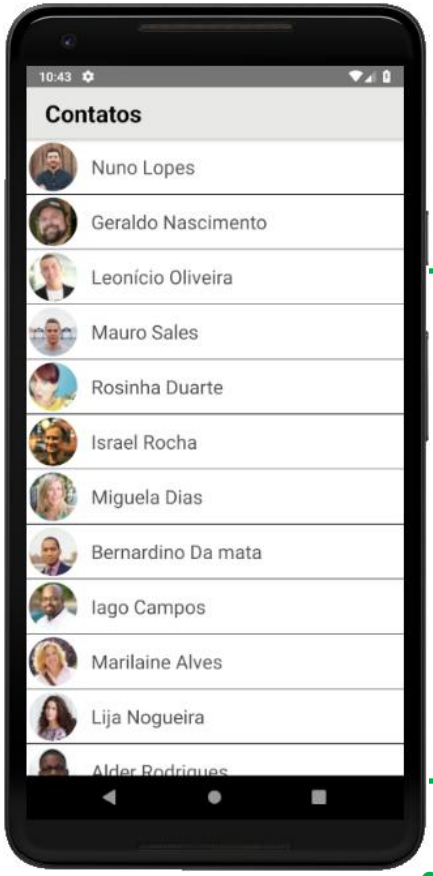
## Configuração/Options *(App.js)*

```
{
  navigationOptions : {
    title : 'Contatos',
    headerTintColor : 'black',
    headerStyle : {
      backgroundColor : '#E8E8E7',
    },
    headerTitleStyle : {
      fontSize : 26,
      color : 'black',
      alignSelf: 'center',
      paddingLeft: 5,
    }
  }
}
```

O *Stack Navigator* permite efetuarmos várias configurações relacionadas aos dados (título) apresentados na barra superior do componentes, bem como a configuração da sua aparência. No exemplo estamos trabalhando com as propriedades *"navigationOption"* + *"headerStyle"* para definir o texto do título (*title*) que será apresentado, a cor (*headerTintColor*) do ícone voltar e a cor de fundo (*backgroundColor*) da barra superior. Além disso, foi utilizada, também, a propriedade *"headerTitleStyle"* para personalizar o texto do título das telas.

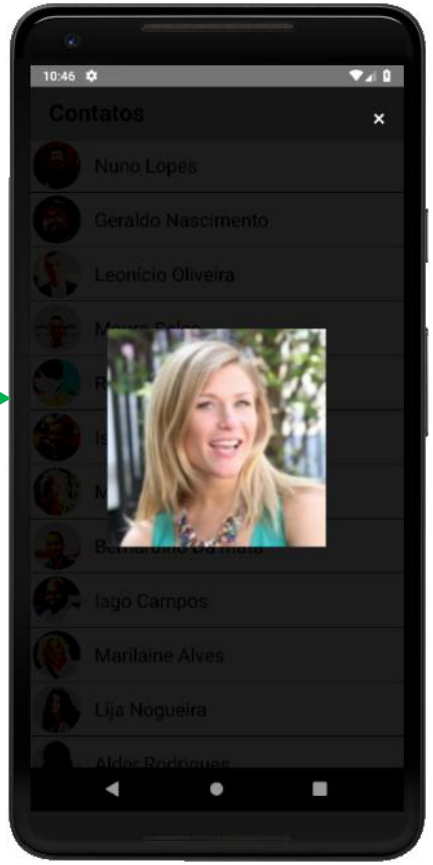
# APP: CONTATOS

## Layout/Funcionamento:

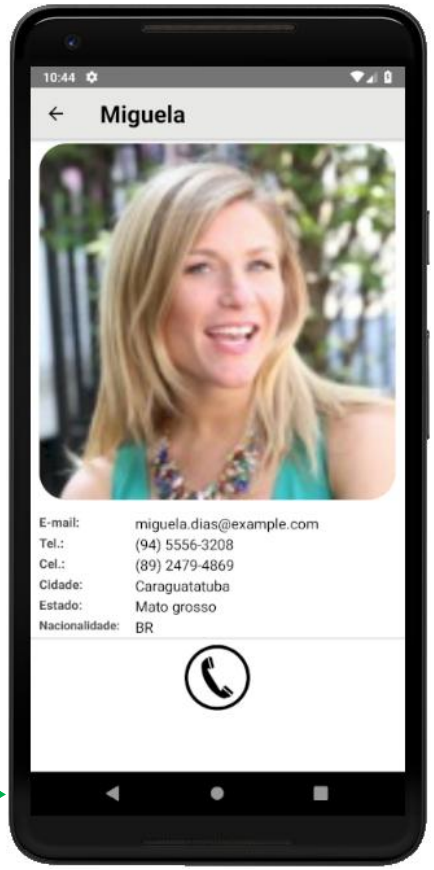



Tela Inicial

Ao clicar na imagem do contato na lista, exibe o *Image View*.




Ao clicar no nome do contato navega para tela que contém informações detalhadas e possibilita efetuar chamada de voz.




Tela Contato

# APP CONTATOS – NAVIGATE

## Navegação/Parâmetros *(ListaContatos.js)*

```
navigate(pessoa) {
  this.props.navigation.navigate('Contato', {pessoa});
}
```

O método ***navigate()*** efetua a navegação para tela de Contato, passando como parâmetro os dados da pessoa escolhida. Observe que a ***props navigatio.navigate*** utilizada, é embutida automaticamente, pelo ***Stack Navigator,*** em todas as telas configuradas anteriormente.

```
renderList() {
  return (
    <ItemList
      itens={this.state.contatos}
      onPress={ (pessoa) => { this.navigate(pessoa) }}
      onPressImg={(pessoa) => { this.onPress(pessoa) }}
    />
  );
}
```

Ao apresentar a Lista de Contatos, o componente ***<ItemList/>*** recebe através da ***props onPress()*** o método ***navigate()***, que será invocado no momento que um contato for selecionado pelo usuário.

```
render() {
  return (
    <View style={styles.container}>
      <View style={styles.list}>
        { this.renderList() }
      </View>
    </View>
  );
}
```

O método ***renderList()*** define o que será exibido: ***Activity Indicator*** ou ***Lista de Contatos***

# APP CONTATOS – NAVIGATE

## Navegação *(Item.js)*

```
<View style={style.view_item}>
    <TouchableOpacity onPress={ () => { props.onPressImg(props.pessoa) } }>
        <Image style={style.avatar} source={{ uri : props.pessoa.picture.thumbnail }} />
    </TouchableOpacity>
    <View style={style.text}>
        <TouchableOpacity onPress={ () => { props.onPress(props.pessoa) } }>
            <Text style={{fontSize: 21}}>
                {UpperCase(props.pessoa.name.first)} {UpperCase(props.pessoa.name.last)}
            </Text>
        </TouchableOpacity>
    </View>

</View>
```

O componente *<Item>* permite visualizar apenas foto do contato, quando a imagem do mesmo é selecionada. Para tal a função ***onPressImg()***, recebida via ***props***, é invocada.

O componente '*<Item>* permite, também, navegar para tela *"Contato"*, quando o nome do mesmo é selecionado. Para tal a função ***onPress ()***, recebida via ***props***, é invocada.

# APP CONTATOS – NAVIGATE

## Tela – Contato *(Contato.js)*

```
render() {

  const { pessoa } = this.props.navigation.state.params;

  const phoneNr = {
    number: pessoa.phone,
    prompt: false,
  }
```

```
  <View style={styles.info}>
    <Info label="E-mail:" content={ pessoa.email } />
    <Info label="Tel.:" content={ pessoa.phone } />
    <Info label="Cel.:" content={ pessoa.cell } />
    <Info label="Cidade:" content={ UpperCase(pessoa.location.city) } />
    <Info label="Estado:" content={ UpperCase(pessoa.location.state) } />
    <Info label="Nacionalidade:" content={ pessoa.nat } />
  </View>
  <TouchableOpacity onPress={() => { call(phoneNr).catch(console.error); }}>
    <Image
      source={ require('../img/call_icon.png') }
      style={{ width: 72, height: 72, alignSelf: 'center' }}
    />
  </TouchableOpacity>
```

Os dados do contato, recebidos via ***props***, são obtidos para que seja possível montar a Tela. Uma constante ***"phoneNr"*** é definida contento o número que será discado.

Os dados do contato são apresentados, o botão de chamada é definido para invocar o método ***call()*** do pacote ***react-nateive-phone-call*** instalado através do comando: ***npm install --save react-native-phone-call***.

# APP: CONTATOS – 2ª Parte

## Exemplos  Utilizados no Documento

## Código-fonte do App Exemplo: Contatos

http://www.gileduardo.com.br/ifpr/pp_rn/downloads/pp_rn_exapp10zip

## Exercício sobre o Conteúdo