React Native

Projeto de Interfaces de Dispositivos Móveis

Aula 02 Navegação

Introdução

- A partir de agora, iremos utilizar a versão nativa do Android, para estudar navegação.
- Siga as instruções de instalação da página:
  - https://reactnative.dev/docs/environment-setup
  - Aba: React Native CLI Quickstart

Introdução

- Versão do Node: use a versão estável do Node.js, de preferência a versão v10.x.x.
- Caso esteja com uma versão acima da 10 e queira voltar, use os seguintes comandos (ubuntu):
  - sudo npm cache clean -f
  - sudo npm install -g n
  - sudo n stable

Introdução

- No Android Studio, crie um emulador (ou use o seu próprio celular) como descrito no site https://reactnative.dev/docs/environment-setup
- Execute o emulador, mas não é necessário manter o Android Studio aberto.

Introdução

- Crie um projeto usando o react-native:
  - npx react-native init aula_navegacao
- Execute o projeto (com o emulador aberto)
  - npx react-native start (em um terminal, dentro do projeto)
  - npx react-native run-android (em outro terminal, dentro do projeto)

Navegação - Stack

- Primeiro, você deve instalar as bibliotecas necessárias:
  - npm install react-navigation --save
  - npm install react-navigation-stack --save
- https://reactnavigation.org/docs/en/hello-react-navigation.html

Navegação - Stack

- Instale também as bibliotecas:
  - npm install react-native-gesture-handler --save
- Faça o link da sua aplicação com a biblioteca anterior (**apenas iOS**):
  - react-native link react-native-gesture-handler

Navegação - Stack

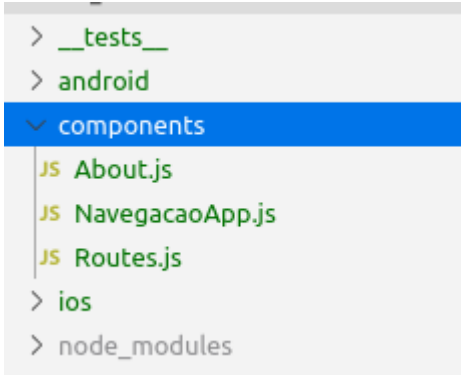
- Instale também as bibliotecas (update 2021):
  - react-native-gesture-handler
  - react-native-reanimated
  - react-native-safe-area-context
  - react-native-screens
  - @react-native-community/masked-view

Navegação - Stack

- Navegação simples: iremos criar dois componentes: o Home e o About.
- Através de um outro componente de rotas, iremos criar os “links” entre as páginas.

Navegação - Stack

- Crie a pasta components e os arquivos:



Navegação - Stack

- NavegacaoApp.js será a página Home.
- About.js será a página About.
- Routes.js organiza as rotas que são usadas na aplicação.

Navegação - Stack

- About.js

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

export default class About extends Component {

    render() {
        return (
            <View style={{ flex: 1, justifyContent: 'center', alignItems: 'center' }}>
                <Text style={{fontWeight:"bold",fontSize:20}}>About</Text>
            </View>
        )
    }
}
```

Navegação - Stack

- NavegacaoApp.js

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

export default class About extends Component {

    render() {
        return (
            <View style={{ flex: 1, justifyContent: 'center', alignItems: 'center' }}>
                <Text>Home ;D</Text>
                <Button
                    title="Ir para About"
                    onPress={() => this.props.navigation.navigate('About')}
                />
            </View>
        )
    }
}
```

Navegação - Stack

- Routes.js

```
import NavegacaoApp from './NavegacaoApp';
import About from './About';

import { createAppContainer } from 'react-navigation';
import { createStackNavigator } from 'react-navigation-stack';

const Routes = createAppContainer(
    createStackNavigator({
        Home: NavegacaoApp,
        About: About,
    },
    {   initialRouteName:'Home'}
    )
);

export default Routes;
```

Navegação - Stack

- App.js

```
import React, {Component} from 'react';

import { SafeAreaProvider } from 'react-native-safe-area-context';
import Routes from './components/Routes'

export default class App extends Component{
  render(){
    return(
      <SafeAreaProvider>
        <Routes/>
      </SafeAreaProvider>
    )
  }
}
```

Navegação - Stack

- **Push** e **GoBack**
  - Funções simples do props.navigation.
  - O Push adiciona mais uma nova página na pilha.
  - O GoBack retorna o histórico.

Navegação - Stack

```
//Em About.js
...
<Button
    title="Ir pro About... de novo?"
    onPress={() => this.props.navigation.navigate('About')}
 />

 <View style={{ padding: 15 }}></View>

 <Button
    title="Ir pro About... de novo!"
    onPress={() => this.props.navigation.push('About')}
 />

 <View style={{ padding: 15 }}></View>

 <Button
    title="Back"
    onPress={() => this.props.navigation.goBack()}
 />
...
```

Navegação - Stack

- Passando parâmetros para rotas:
  - Primeiro, usa-se o this.props.navigation.navigate para passar um objeto de parâmetros:
    - this.props.navigation.navigate('RouteName', { /* params go here */ })
  - Segundo, pega-se os parâmetros na página destino:
    - this.props.navigation.getParam(paramName, defaultValue)

Navegação - Stack

NavegacaoApp.js (render)

```
...
<Button title="Ir para About"
        onPress={() => this.props.navigation.navigate('About',{nome:'Jefferson'})}
         />
...
```

About.js (render)

```
...
    const nome = this.props.navigation.getParam('nome','nulo');
     return (
        <View style={{ flex: 1, justifyContent: 'center', alignItems: 'center' }}>
          <Text style={{ fontWeight: "bold", fontSize: 20 }}>About: {nome}</Text>
...
```

Navegação - Stack

- Configurando o cabeçalho (header):

```
...
export default class About extends Component {

    static navigationOptions = {
        title: 'Home',
        headerStyle: {
            backgroundColor: '#f4511e',
        },
        headerTintColor: '#fff',
        headerTitleStyle: {
            fontWeight: 'bold',
        },
    };
...
```

Navegação - Stack

```
const Routes = createAppContainer(

    createStackNavigator(
        {
            Home: NavegacaoApp,
            About: About,
        },
        {
            initialRouteName: 'Home',
            defaultNavigationOptions: {
                headerStyle: {
                    backgroundColor: '#f4511e',
                },
                headerTintColor: '#fff',
                headerTitleStyle: {
                    fontWeight: 'bold',
                },
            },
        }
    )
);
```

Outra opção é colocar no Routes.js, dentro da pilha (stack) de páginas a qual você quer que todas as páginas tenham o mesmo estilo de header.

Navegação - Stack

- JSX no header (nesse caso, use a propriedade “headertitle”):

```
static navigationOptions = {
        //title: 'About',
        headerTitle: <Text style={{fontWeight:"bold", color:"black", fontSize:20}}>About in JSX</Text>
        /*headerStyle: {
            backgroundColor: '#f4511e',
        },
        headerTintColor: '#fff',
        headerTitleStyle: {
            fontWeight: 'bold',
        },*/
    };
```

Navegação - Stack

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

export default class Logo extends Component {
    render() {
        return (
            <View style={{flexDirection:"row"}}>
                <Image
                    source={require('./jeff.png')}
                    style={{ width: 30, height: 30 }}
                />
                <Text>{this.props.titulo}</Text>
            </View>
        )
    }
}
```

Arquivo Logo.js, dentro de /components. Outra opção, é criar um componente a parte para chamar em headerTitle:

```
headerTitle: <Logo titulo='About'/>
```

Navegação - Stack

- Botões no cabeçalho...

```
static navigationOptions = {
        headerTitle: <Logo titulo='About' />,
        headerRight: (
            <Button
                onPress={() => alert('Um botão!')}
                title="Info"
                color="#000"
            />
        ),
}
```

```
export default class About extends Component {

    constructor(props) {
        super(props);
        this.state = { contador: 0 };
    }

    aumentarContador = () => {
        this.setState({ contador: this.state.contador + 1 });
    }

    componentDidMount() {
        this.props.navigation.setParams({ aumentarContador: this.aumentarContador });
    }

    static navigationOptions = ({ navigation }) => {
        return {
            headerTitle: <Logo titulo='About' />,
            headerRight: (
                <Button
                    onPress={navigation.getParam('aumentarContador')}
                    title="+1"
                    color="#000"
                />
            ),
        };
    };
...
```

Nesse exemplo, um botão no cabeçalho mofica variáveis no componente central.

Perceba que o navigationOptions teve que ser transformado em um método, para poder acessar o objeto “navigation”...

Navegação - Stack

- **Modais**: janelas que se sobrepõem sobre o fluxo principal da aplicação, necessitando de uma resposta do usuário para sumam. Semelhantes aos pop-ups.

Navegação - Stack

- Antes de mais nada, vamos criar um arquivo chamada Modal.js, dentro na nossa pasta “components” e nele criaremos o componente “Modal”.

Navegação - Stack

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

export default class Modal extends Component {
    render() {
        return (
            <View style={{ flex: 1, alignItems: 'center', justifyContent: 'center' }}>
                <Text style={{ fontSize: 30 }}>Isto é um Modal!</Text>
                <Button
                    onPress={() => this.props.navigation.goBack()}
                    title="Dismiss"
                />
            </View>
        );
    }
}
```

Navegação - Stack

- Devemos alterar o arquivo Routes.js para criar dois tipos de pilha de navegação:
  - Uma para a navegação principal, ou seja, a que já estamos trabalhando originalmente. Será chamada de **MainStack**.
  - Uma outra para incluirmos o Modal e a Pilha de navegação principal. Chamaremos ela de **RootStack**.

Navegação - Stack

Routes.js (snippet)

```
const MainStack = createStackNavigator(
    {
        Home: NavegacaoApp,
        About: About,
    },
    {
        initialRouteName: 'Home',
        defaultNavigationOptions: {
            headerStyle: {
                backgroundColor: '#f4511e',
            },
            headerTintColor: '#fff',
            headerTitleStyle: {
                fontWeight: 'bold',
            },
        },
    }
);
```

A nossa pilha original, apenas a colocamos em uma variável separada...

```
const RootStack = createStackNavigator(
    {
        Main:MainStack,
        MyModal: Modal,
    },
    {
        mode: 'modal',
        headerMode: 'none'
    }
)

const Routes =
createAppContainer(RootStack);
```

A nova pilha raiz. Inclui a principal e o componente Modal. A novidade é o mode: 'modal'. No entanto, a propriedade Main coloca a pilha anterior como principal.

Navegação - Stack

- Finalmente, crie na página Home (NavegacaoApp.js) um botão para chamar o Modal:

```
 ...
<View style={{ padding: 20 }}>
    <Button
      title="Abrir Modal"
      onPress={() => this.props.navigation.navigate('MyModal')}
    />
 </View>
...
```

Navegação - Tabs

- Possibly the most common style of navigation in mobile apps is tab-based navigation. This can be tabs on the bottom of the screen or on the top below the header (or even instead of a header).

Navegação - Tabs

- Preparando o ambiente:
  - npm install react-navigation-tabs --save
  - npm install react-native-reanimated --save
  - react-native link react-native-reanimated (apenas iOS)

Navegação - Tabs

- Dentro da pasta components:
  - cria uma outra pasta chamada “tab”
  - crie o arquivo “TabSimples.js”
  - O código fonte será mais simplificado...vejamos no próximo slide.

Navegação - Tabs

```
import React from 'react';
import { Text, View } from 'react-native';
import { createAppContainer } from 'react-navigation';
import { createBottomTabNavigator } from 'react-navigation-tabs';

class HomeScreen extends React.Component {
  render() {
    return (
      <View style={{ flex: 1, justifyContent: 'center', alignItems: 'center' }}>
        <Text>Home!</Text>
      </View>
    );
  }
}

class SettingsScreen extends React.Component {
  render() {
    return (
      <View style={{ flex: 1, justifyContent: 'center', alignItems: 'center' }}>
        <Text>Settings!</Text>
      </View>
    );
  }
}

const TabNavigator = createBottomTabNavigator({
  Home: HomeScreen,
  Settings: SettingsScreen,
});

export default createAppContainer(TabNavigator);
```

Navegação - Tabs

- Estilo na Tab:

```
...
const TabNavigator = createBottomTabNavigator(
    {
        Home: HomeScreen,
        Settings: SettingsScreen,
    },
    {
        tabBarOptions: {
            activeTintColor: 'tomato',
            inactiveTintColor: 'gray',
            activeBackgroundColor: 'yellow',
            labelStyle: {
                fontSize: 20,
                paddingBottom: 10,
                fontWeight: "bold"
            },
        },
    }
);
...
```

Navegação - Tabs

- Pulando entre telas:
  - Podemos também adicionar botões nas nossas telas para irmos para outras tabs!
  - Crie um novo arquivo chamada JumpingTabs.js. Seu código fonte estará no próximo slide.

Navegação - Tabs

```
import React from 'react';
import { Button, Text, View } from 'react-native';
import { createAppContainer } from 'react-navigation';
import { createBottomTabNavigator } from 'react-navigation-tabs';

class HomeScreen extends React.Component {
  render() {
    return (
      <View style={{ flex: 1, justifyContent: 'center', alignItems: 'center' }}>
        <Text>Home!</Text>
        <Button
          title="Go to Settings"
          onPress={() => this.props.navigation.navigate('Settings')}
        />
      </View>
    );
  }
}
```

```
class SettingsScreen extends React.Component {
  render() {
    return (
      <View style={{ flex: 1, justifyContent: 'center', alignItems: 'center' }}>
        <Text>Settings!</Text>
        <Button
          title="Go to Home"
          onPress={() => this.props.navigation.navigate('Home')}
        />
      </View>
    );
  }
}

export default createAppContainer(
  createBottomTabNavigator(
    {
      Home: HomeScreen,
      Settings: SettingsScreen,
    },
  ));
```

Navegação - Drawer

- Navegação no estilo “gavetas”, que podem ser escondidas nos cantos da tela.
- Para os gestos funcionarem, deve-se instalar:
  - npm install react-native-gesture-handler --save
  - npm install 'react-navigation-drawer --save
- Crie o arquivo DrawerSimples.js

Navegação - Drawer

- Imports:

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

import { createAppContainer } from 'react-navigation';
import { createDrawerNavigator } from 'react-navigation-drawer';
```

Navegação – Drawer (Tela Home)

```
class MyHomeScreen extends React.Component {

  static navigationOptions = {
    drawerLabel: 'Home',
    drawerIcon: ({ tintColor }) => (
      <Image
        source={require('./icon-chat.png')}
        style={[styles.icon, { tintColor: tintColor }]}
      />
    ),
  };

  render() {
    return (
      <View style={styles.container}>

        <Text style={styles.title}> HOME SCREEN </Text>

        <Text
          style={styles.paragraph}
          onPress={() => {
            this.props.navigation.navigate('Notifications');
          }}>
          Go to Notifications
        </Text>

        <Text
          style={styles.paragraph}
          onPress={() => {
            this.props.navigation.toggleDrawer();
          }}>
          Toggle Drawer
        </Text>
      </View>
    );
  }
}
```

Baixe alguns ícones...(está no nosso GIT)

Link simples que leva direto para a página de notificações.

Abre e fecha gaveta lateral.

Navegação – Drawer (Tela Notifications)

```
class MyNotificationsScreen extends React.Component {
  static navigationOptions = {
    drawerLabel: 'Notifications',
    drawerIcon: ({ tintColor }) => (
      <Image
        source={require('./icon-notify.png')}
        style={[styles.icon, { tintColor: tintColor }]}
      />
    ),
  };

  render() {
    return (
      <View style={styles.container}>
        <Text style={styles.title}> NOTIFICATIONS SCREEN </Text>
        <Text
          style={styles.paragraph}
          onPress={() => {
            this.props.navigation.navigate('Home');
          }}>
          Go back home
        </Text>
      </View>
    );
  }
}
```

Navegação – Drawer (Estilos e Navegador)

```
const styles = StyleSheet.create({
    icon: {
        width: 24,
        height: 24,
    },
    container: {
        flex: 1,
        justifyContent: 'center',
        paddingTop: 5,
        backgroundColor: '#ecf0f1',
        padding: 8,
    },
    paragraph: {
        margin: 24,
        fontSize: 18,
        fontWeight: 'bold',
        textAlign: 'center',
    },
    title: {
        margin: 15,
        fontSize: 25,
        fontWeight: 'bold',
        textAlign: 'center',
    },
});
```

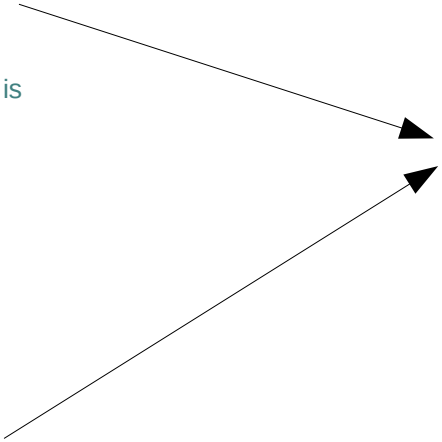
```
const MyDrawerNavigator = createDrawerNavigator({
    Home:MyHomeScreen,
    Notifications: MyNotificationsScreen
});

export default createAppContainer(MyDrawerNavigator);
```

Navegação - Drawer

- Para os gestos funcionarem no Android, é necessário fazer uma alteração no arquivo:
- <projeto>/android/app/src/main/java/com/aula_naveg/**MainActivity.java**

Navegação – Drawer (MainActivity.java)

```
package com.aula_naveg;

import com.facebook.react.ReactActivity;

import com.facebook.react.ReactActivityDelegate;
import com.facebook.react.ReactRootView;
import com.swmansion.gesturehandler.react.RNGestureHandlerEnabledRootView;

public class MainActivity extends ReactActivity {

    /**
     * Returns the name of the main component registered from JavaScript. This is
     * used to schedule rendering of the component.
     */
    @Override
    protected String getMainComponentName() {
        return "aula_naveg";
    }

    @Override
    protected ReactActivityDelegate createReactActivityDelegate() {
        return new ReactActivityDelegate(this, getMainComponentName()) {
            @Override
            protected ReactRootView createRootView() {
                return new RNGestureHandlerEnabledRootView(MainActivity.this);
            }
        };
    }
}
```



Código novo.

Navegação Switch

- Mostrar apenas uma tela por vez que, por padrão, não permite a operação de “voltar”.
- Em uma aplicação de autenticação, por exemplo, nós teremos as telas de autenticação em uma pilha separada (login, lembrar senha, sign in). Uma vez autenticado, não queremos mais voltar para as telas de autenticação ao pressionar em “back”.

Navegação Switch (Routes.js)

```
import { createAppContainer, createSwitchNavigator } from 'react-navigation';
import { createStackNavigator } from 'react-navigation-stack';


import HomeScreen from './HomeScreen';
import OtherScreen from './OtherScreen';
import SignInScreen from './SignInScreen';
import AuthLoadingScreen from './AuthLoadingScreen'

const AppStack = createStackNavigator({ Home: HomeScreen, Other: OtherScreen });
const AuthStack = createStackNavigator({ SignIn: SignInScreen });

export default createAppContainer(createSwitchNavigator(
  {
    AuthLoading: AuthLoadingScreen,
    App: AppStack,
    Auth: AuthStack,
  },
  {
    initialRouteName: 'AuthLoading',
  }
));
```

Navegação Switch (AuthLoading.js)

```
import React from 'react';
import { ActivityIndicator, AsyncStorage,
          StatusBar, StyleSheet, View } from 'react-native';

export default class AuthLoadingScreen extends React.Component {
    componentDidMount() {
        this._bootstrapAsync();
    }


    _bootstrapAsync = async () => {
        const userToken = await AsyncStorage.getItem('userToken');
        this.props.navigation.navigate(userToken ? 'App' : 'Auth');
    };

    // Render any loading content that you like here
    render() {
        return (
            <View style={styles.container}>
                <ActivityIndicator />
                <StatusBar barStyle="default" />
            </View>
        );
    }
}
```

```
const styles = StyleSheet.create({
    container: {
        flex: 1,
        alignItems: 'center',
        justifyContent: 'center',
    },
});
```

Navegação Switch (SignInScreen.js)

```
import React from 'react';
import { AsyncStorage, View, StyleSheet, Button} from 'react-native';

export default class SignInScreen extends React.Component {
    static navigationOptions = {
        title: 'Please sign in',
    };

    render() {
        return (
            <View style={styles.container}>
                <Button title="Sign in!" onPress={this._signInAsync} />
            </View>
        );
    }

    _signInAsync = async () => {
        await AsyncStorage.setItem('userToken', 'abc');
        this.props.navigation.navigate('App');
    };
}
```

Navegação Switch (HomeScreen.js)

```
import React from 'react';
import { AsyncStorage, View,
          StyleSheet, Button} from 'react-native';

export default class HomeScreen extends React.Component {
   static navigationOptions = {
      title: 'Welcome to the app!',
   };

   render() {
      return (
         <View style={styles.container}>
            <Button title="Show me more of the app" onPress={this._showMoreApp} />
            <View style={{padding:15}}></View>
            <Button title="Actually, sign me out :)" onPress={this._signOutAsync} />
         </View>
      );
   }

   _showMoreApp = () => {
      this.props.navigation.navigate('Other');
   };

   _signOutAsync = async () => {
      await AsyncStorage.clear();
      this.props.navigation.navigate('Auth');
   };
}
```

Navegação Switch (OtherScreen.js)

```
import React from 'react';
import { AsyncStorage, StatusBar,
          View, StyleSheet, Button} from 'react-native';

export default class OtherScreen extends React.Component {
   static navigationOptions = {
      title: 'Lots of features here',
   };

   render() {
      return (
         <View style={styles.container}>
            <Button title="I'm done, sign me out" onPress={this._signOutAsync} />
            <StatusBar barStyle="default" />
         </View>
      );
   }

   _signOutAsync = async () => {
      await AsyncStorage.clear();
      this.props.navigation.navigate('Auth');
   };
}
```

AsyncStorage (eliminar Warning)

- Install AsyncStorage
  - Install it using your favourite package manager npm or yarn
  - Link the dependency
  - Use the dependency
- Installation: choose the method you usually use
  - npm i @react-native-community/async-storage
- Link the dependency (you may not have to do this if you are using 0.60+ as it has Autolinking)
  - react-native link @react-native-community/async-storage
- Then you import it like this, and use it as before.
  - import AsyncStorage from '@react-native-community/async-storage';