import React, { useState } from "react";

import { createNativeStackNavigator } from "@react-navigation/native-stack";

import TelaComPush from "../views/TelaComPush";

import PassoStackComPush from "../componentes/PassoStackComPush";

import TelaInitialPush from "../views/TelaInitialPush";

/\*  MÉTODO PUSH:

    No Stack podemos contar com 2 métodos para ir para uma próxima tela, podemos contar como método "navigate" e

    o método "push". Aparentemente parecem que os 2 fazem a mesma coisa: "ir para a próxima tela", mas não,

    existem diferenças muito importantes entre estes médotos, e essas diferenças devem ser consideradas:

        \* Método navigate:

            Esse método faz com que avancemos para uma próxima tela. Porém, digamos que desejamos que ao apertar

            um botão para a mesma tela, onde, queremos renderizar a mesma tela, porém com um resultado

            diferente. Isso seria impossível com o "navigate", ele não chama a mesma tela!

        \* Métoso push:

            O método "push", faz a mesma coisa que o "navigate", com a diferença de que ele consegue renderizar

            a mesma tela. Isso é muito último quando queremos usar o mesmo estilo de layout, porém mostrar

            conteúdo diferente.

    Vejamos um exemplo...

\*/

const Stack = createNativeStackNavigator()

/\*No exemplo abaixo temos um componente que vai chamar a mesma tela, porém, a cada chamada de tela ele mostra

quantas telas nós avançamos. Além disso, de apertarmos no voltar, ele mostra quantas telas nós retornamos...

\*/

export default props => {

    //Como vamos trabalhar com renderização, vamos usar hooks...

    const [num, setNum] = useState(0)

    const addNum = () => setNum(num + 1)

    const minusNum = () => setNum(num - 1)

    return (

        <Stack.Navigator initialRouteName="TelaInitialPush" screenOptions={{headerShown: false}}>

            <Stack.Screen name='TelaInitialPush'>

                {props => (

                    <PassoStackComPush {...props} avancar="TelaComNumeros" funcoes={[addNum]}>

                        <TelaInitialPush enum={num} />

                    </PassoStackComPush>

                )}

            </Stack.Screen>

            <Stack.Screen name='TelaComNumeros'>

                {/\* Note aqui que estamos usando uma tela para chamar ela mesma... \*/}

                {props => (

                    <PassoStackComPush {...props} avancar="TelaComNumeros" voltar funcoes={[addNum, minusNum]}>

                        <TelaComPush enum={num} />

                    </PassoStackComPush>

                )}

            </Stack.Screen>

        </Stack.Navigator>

    )

}

ARQUIVO PASSOSTACKCOMPUSH...

import React from "react";

import { View, StyleSheet, Button } from "react-native";

export default props => {

    // Aqui utilizamos as funções para somar um valor ao hook ou retirar o valor

    const avancar = tela => {

        props.navigation.push(tela)

        props.funcoes[0]()

    }

    const voltar = () => {

        props.navigation.goBack()

        props.funcoes[1]()

    }

    return (

        <View style={styles.flexInteiro}>

            <View>

                {props.voltar ? <Button title="Voltar" onPress={voltar} /> : false}

                {props.avancar ? <Button title="Avançar" onPress={() => avancar(props.avancar)} /> : false}

            </View>

            <View style={styles.flexInteiro}>

                {props.children}

            </View>

        </View>

    )

}

const styles = StyleSheet.create({

    flexInteiro: {

        flex: 1,

    },

})

ARQUIVO TELA INITIAL...

import React from "react";

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

export default props => {

    // Essa é somente uma tela inicial, onde explicamos a proposta e não vamos ter um botão de voltar...

    return (

        <View style={styles.view}>

            <Text style={styles.text}>Tela Inicial</Text>

            <Text style={styles.text}>Avance pelas outras pelas e veja abaixo a contagem das vezes que você avançou, se você voltar, a contagem será regredida.</Text>

            <Text style={styles.textNumbs}>{props.enum}</Text>

        </View>

    )

}

const styles = StyleSheet.create({

    view: {

        alignItems: 'center',

        flex: 1,

        justifyContent: 'center',

        padding: 15,

    },

    text: {

        fontSize: 25,

        textAlign: "center",

    },

    textNumbs: {

        color: 'black',

        fontSize: 50,

    },

})

ARQUIVO TELACOMPUSH....

import React from "react";

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

export default props => {

    return (

        <View style={styles.view}>

            <Text style={styles.text}>Tela dos Números</Text>

            <Text style={styles.textNumbs}>{props.enum}</Text>

        </View>

    )

}

const styles = StyleSheet.create({

    view: {

        alignItems: 'center',

        flex: 1,

        justifyContent: 'center',

    },

    text: {

        fontSize: 25,

    },

    textNumbs: {

        color: 'black',

        fontSize: 50,

    },

})

ARQUIVO INDEX.JS...

import React from "react";

import { SafeAreaView, StyleSheet } from "react-native";

//Aqui temos a importação do componente NavigationContainer...

import { NavigationContainer } from "@react-navigation/native";

import StackComPush from "./StackComPush";

export default props => {

    return (

        <SafeAreaView style={styles.safeArea}>

            {/\* Note que o NavigationContainer comporta a nossa Stack... \*/}

            <NavigationContainer>

                <StackComPush />

            </NavigationContainer>

        </SafeAreaView>

    )

}

const styles = StyleSheet.create({

    safeArea: {

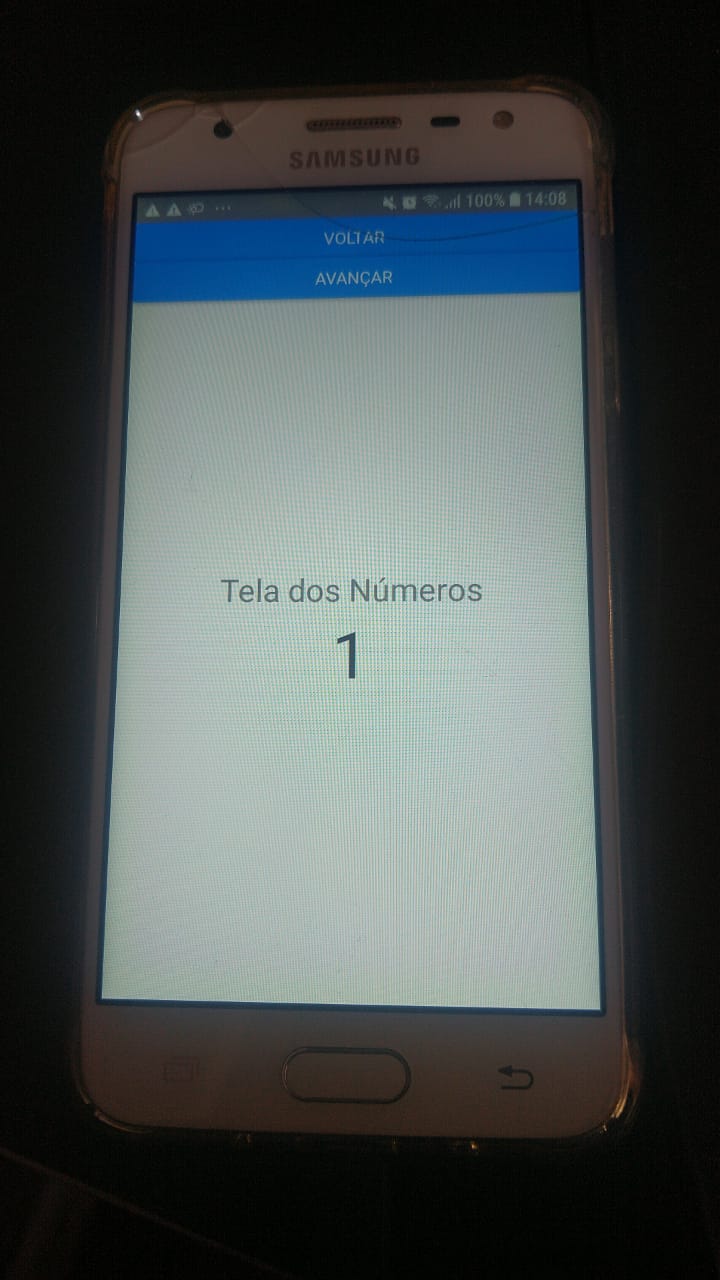
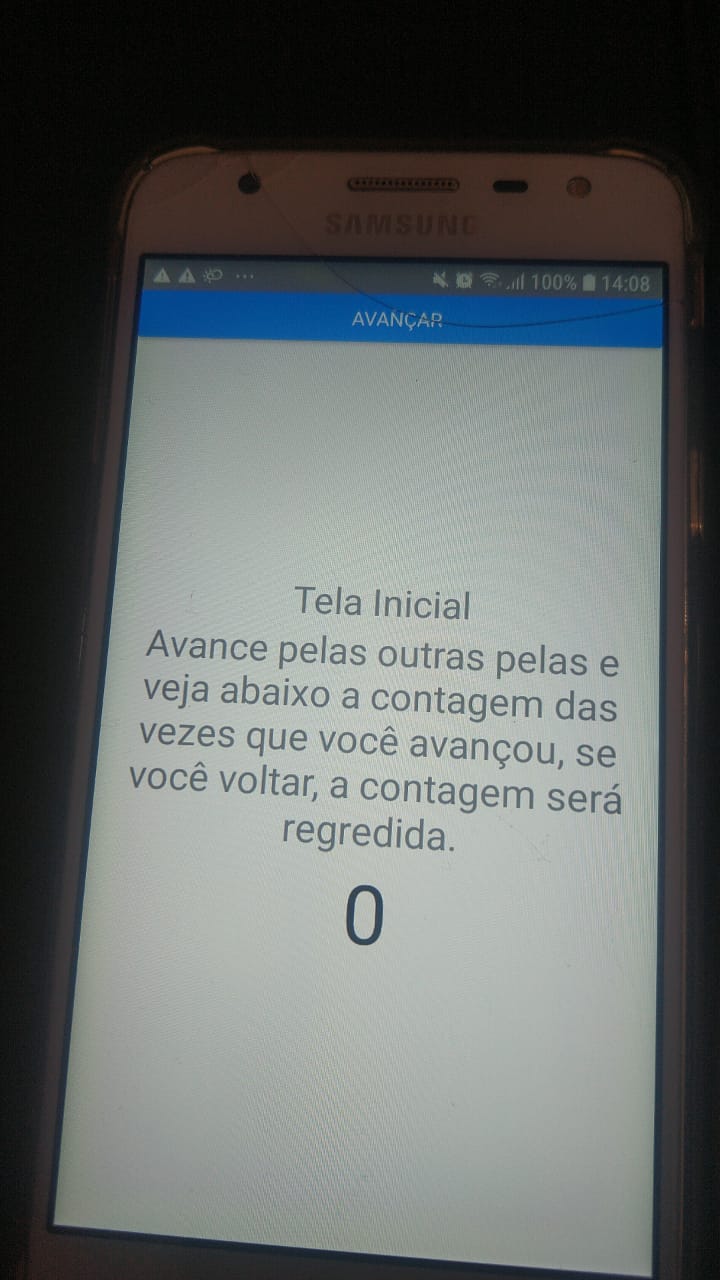
        flex: 1,

    },

})

RESULTADOS NO ANDROID...

CONTINUA....



RESULTADO DEPOIS DE AVANÇAR MAIS 3 VEZES, E DEPOIS, DEPOIS DE VOLTAR 2 VEZES...

