Programação para Dispositivos Móveis
Aula 4

Prof: Uemerson Pinheiro Junior

Trabalhando com animações

- Iremos trabalhar com Animated
- Iremos trabalhar com Easing
- Função recursiva

Passo 1

- Entre no site: https://snack.expo.dev/
- Ache o projeto que fizemos na aula passada (heads-or-tails)

Passo 2

- Remova todos os imports e adicione os seguintes:

```
import React, { useState, useRef, useEffect } from 'react';
import {
  SafeAreaView,
  StyleSheet,
  Image,
  TouchableOpacity,
  Text,
  View,
  Animated,
  Easing,
} from 'react-native';
```

Passo 3

- Depois dos estados coinSide, headsCount e tailsCount, adicione o código a seguir:

```
const flipAnimation = useRef(new Animated.Value(0)).current;
const [updateCoinCount, setUpdateCoinCount] = useState( false);
const [disabledButton, setDisabledButton] = useState( false);
```

Passo 4

- Depois da constante **updateCoinCount**, adicione o código a seguir:

```
  useEffect(() => {
    if (updateCoinCount) {
      if (coinSide === 'Heads') setHeadsCount(headsCount + 1);
      else setTailsCount(tailsCount + 1);
      setUpdateCoinCount(false);
      setDisabledButton(false);
    }
  }, [coinSide, updateCoinCount, headsCount, tailsCount]);
```

Passo 5

- Depois do **useEffect**, adicione o código a seguir:

```
  const flipCoin = (randomFlips) => {
    flipAnimation.setValue(0);
    setDisabledButton(true);

    Animated.timing(flipAnimation, {
      toValue: 1,
      duration: 500,
      easing: Easing.linear,
      useNativeDriver: true,
    }).start(() => {
      setCoinSide((prevCoinSide) =>
        prevCoinSide === 'Heads' ? 'Tails' : 'Heads'
      );
      if (randomFlips > 0) {
        flipCoin(randomFlips - 1);
      } else setUpdateCoinCount(true);
    });
  };
```

Passo 6.0

- Remova completamente o componente <Image />, que contém as imagens das moedas:

```
      <Image
        style={styles.coin}
        source={
          coinSide === 'Heads'
            ? require('./assets/heads.png')
            : require('./assets/tails.png')
        }
      />
```

Passo 6.1

- No lugar que removemos o componente <Image />, adicione o código a seguir:

```
<View style={styles.coinContainer}>
        {coinSide && (
          <Animated.Image
            source={
              coinSide === 'Heads'
                ? require('./assets/heads.png')
                : require('./assets/tails.png')
            }
```

Passo 6.2

- Continue adicionando:

```
    style={[
              styles.coin,
              {
                transform: [
                  {
                    rotateY: flipAnimation.interpolate({
                      inputRange: [0, 1],
                      outputRange: ['0deg', '180deg'],
                    }),
                  },
                ],
              },
            ]}
          />
        )}
      </View>
```

Passo 7.0

- Remova o estilo e a função onPress do componente <TouchableOpacity> a seguir:

```
style={styles.button}
onPress={() => {
    const randomCoin = Math.floor(Math.random() * 2);
    setCoinSide(randomCoin == 0 ? 'Heads' : 'Tails');

    if (coinSide === 'Heads') setHeadsCount(headsCount + 1);
    else setTailsCount(tailsCount + 1);
}}>
```

Passo 7.1

- No lugar adicione a função o novo estilo, a função **onPress** e a propriedade disabled a seguir:

```
style={{ ...styles.button, opacity: disabledButton ? 0.1 : 1 }}
onPress={() => flipCoin(Math.floor(Math.random() * 10) + 1)}
disabled={disabledButton}
```

Passo 8

- Adicione o estilo no final em **styles**:

```
  coinContainer: {
    marginBottom: 30,
  },
```

Agora vamos entender o que fizemos!

Exercícios:

1. Altere as imagens cara e coroa por outras imagens. Sugestão: Uma foto sua e de um amigo(a)
2. Para fazer sentido com as novas imagens mude os texto de cara ou coroa
3. Para fazer sentido com as novas imagens coloque uma logo nova no app