Vitor Amadeu Souza

Introdução ao

React Native

No ambiente Expo Snack

Parte X

Outubro de 2024

Direitos reservados por:
Cerne Tecnologia e Treinamento Ltda
Produção: Cerne Tecnologia e Treinamento
E-mail da Empresa: cerne@cerne-tec.com.br
Home Page: www.cerne-tec.com.br.com.br
Atendimento ao Consumidor: sac@cerne-tec.com.br
Contato com o Autor: vitor@cerne-tec.com.br

FEITO NO BRASIL

“Venha também sobre mim a tua benignidade, ó Senhor, e a tua salvação, segundo a tua palavra.”

Sl 119:41

Cerne Tecnologia

A Cerne Tecnologia tem uma equipe preparada para desenvolvimento de projetos eletrônicos em diversas áreas: Médica, Entretenimento, Industrial, Robótica, Científica, Automobilística, Aeronáutica, etc. Trabalhamos com tecnologia microcontrolada usando o PIC, ARM, AVR, 8051, dsPIC, PIC24, PIC32 além do Arduino, Raspberry, Beaglebone etc. Desenvolvemos o projeto desde sua concepção até a entrega do produto final, passando pelas etapas de esquema elétrico, protótipo e desenvolvimento de circuito impresso.

Desenvolvemos aplicativos para smartphones/tablets Android, iOS, Blackberry, Windows Phone e no desenvolvimento de softwares a nível PC para plataforma Windows, usando ferramentas como o Visual Basic, C# e C++.

Atuamos na parte de montagem de placas, onde podemos fornecer ambos os serviços de desenvolvimento de projetos e produção ou apenas um destes.

Desenvolvemos esquemas elétricos e layout de PCI, tanto em tecnologia convencional como SMD.

Temos a flexibilidade de customizar um de nossos produtos, de modo a atender a uma necessidade específica do cliente, tornando o custo de desenvolvimento menor se comparado a construção de um projeto desde a sua fase inicial.

Desenvolvemos e fornecemos kits didáticos para diversos microcontroladores além de apostilas, livros e e-books.

Na hora de desenvolver um projeto ou equipar seu laboratório não hesite em nos contatar. Entre em contato conosco através do endereço cerne-tec.com.br para obter mais informações.

Cerne
Conhecimento para o Desenvolvimento

cerne-tec.com.br

Sumário

Capítulo I
Metodologia de desenvolvimento

1. Introdução

Neste livro, você encontrará uma parte da série dedicada ao desenvolvimento de projetos com React Native. Ao longo desta obra, diversos experimentos serão apresentados conforme o sumário, permitindo um aperfeiçoamento contínuo no uso deste framework para a construção de interfaces responsivas com foco em dispositivos móveis.

Capítulo II
Programação em React Native

1. API de Geolocalização

O código implementa um aplicativo em React Native utilizando o Expo para acessar a localização do dispositivo. Ele faz uso do hook useState para armazenar os dados de localização e as mensagens de erro, e useEffect para executar a função de obtenção de localização ao carregar o aplicativo. A função getLocation solicita permissão para acessar a localização do dispositivo por meio do método requestForegroundPermissionsAsync da biblioteca expo-location. Caso a permissão seja concedida, a função obtém as coordenadas atuais do usuário utilizando o método getCurrentPositionAsync, armazenando os dados no estado.

A interface exibe uma mensagem que é atualizada conforme o status da localização. Inicialmente, a mensagem exibe "Buscando...", mas se houver algum erro, como a negação de permissão, a mensagem de

erro será exibida. Caso a localização seja obtida com sucesso, a latitude e longitude são apresentadas na tela. Além disso, um botão é oferecido para atualizar manualmente a localização a qualquer momento, acionando novamente a função de obtenção de coordenadas.

O uso de useEffect garante que a localização seja buscada assim que o aplicativo é iniciado, e a interface é gerenciada de forma a refletir a atualização de estado. O layout é estilizado utilizando StyleSheet com uma abordagem simples e limpa.

```
import React, { useState, useEffect } from 'react';
import { Text, View, StyleSheet, Button } from 'react-native';
import * as Location from 'expo-location';

export default function App() {
  const [location, setLocation] = useState(null);
  const [errorMsg, setErrorMsg] = useState(null);


  const getLocation = async () => {
    // Solicita permissão para acessar a localização
    let { status } = await Location.requestForegroundPermissionsAsync();
    if (status !== 'granted') {
      setErrorMsg('Permissão para acessar a localização foi negada.');
      return;
    }


    // Obtém a localização atual
    let location = await Location.getCurrentPositionAsync({});
    setLocation(location);
```

```
  };

  useEffect(() => {
    getLocation();
  }, []);

  let text = 'Buscando...';
  if (errorMsg) {
    text = errorMsg;
  } else if (location) {
    text = `Latitude: ${location.coords.latitude}, Longitude: $
{location.coords.longitude}`;
  }

  return (
    <View style={styles.container}>
      <Text style={styles.paragraph}>{text}</Text>
      <Button title="Atualizar localização" onPress={getLocation} />
    </View>
  );
}

const styles = StyleSheet.create({
  container: {
    flex: 1,
    justifyContent: 'center',
    alignItems: 'center',
    paddingTop: 50,
    backgroundColor: '#ecf0f1',
  },
  paragraph: {
    margin: 24,
    fontSize: 18,
    textAlign: 'center',
  },
});
```

O código começa criando dois estados, location e errorMsg, que armazenam, respectivamente, as coordenadas de localização e qualquer mensagem de

erro caso a permissão seja negada. Quando o aplicativo é carregado, a função getLocation é chamada automaticamente por meio do hook useEffect. Essa função solicita permissão ao usuário para acessar a localização, utilizando o método Location.requestForegroundPermissionsAsync. Se a permissão for concedida, a função Location.getCurrentPositionAsync obtém as coordenadas (latitude e longitude) do dispositivo, que são então armazenadas no estado location.

Se o usuário não conceder permissão, uma mensagem de erro é armazenada em errorMsg e exibida na tela. Caso contrário, as coordenadas de latitude e longitude são exibidas. O aplicativo também inclui um botão que permite ao usuário atualizar manualmente a localização, chamando novamente a função getLocation ao ser pressionado. A interface é simples, com uma mensagem de texto que exibe "Buscando..." enquanto a localização está sendo obtida, ou exibe a mensagem de erro ou as coordenadas assim que disponíveis. O layout é organizado de forma