Vitor Amadeu Souza

Introdução ao

React Native

No ambiente Expo Snack

Parte XIV

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. Lista de Tarefas

O código apresentado é uma aplicação de lista de tarefas desenvolvida em React Native, permitindo que os usuários adicionem e removam tarefas, proporcionando uma interface básica e funcional. A estrutura do código começa com a importação das bibliotecas necessárias, incluindo React e useState para gerenciamento de estado, além de componentes do react-native, como View, TextInput, Button, FlatList, Text e StyleSheet.

O componente principal da aplicação é chamado TodoList. Dentro deste componente, dois estados são definidos: o primeiro, task, armazena o texto da tarefa que o usuário deseja adicionar, e o segundo, tasks, é um array que contém todas as tarefas adicionadas. A função addTask é chamada quando o usuário pressiona o botão "Adicionar". Se o campo de entrada não estiver vazio, a nova tarefa é adicionada ao array de tarefas e o campo é limpo. Por outro lado, a função removeTask permite remover uma tarefa específica com base

no índice passado. Ela utiliza o método filter para criar um novo array que exclui a tarefa selecionada e atualiza o estado de tarefas.

A interface do usuário é composta por um campo de entrada (TextInput) onde o usuário pode digitar uma nova tarefa, um botão "Adicionar" que chama a função addTask ao ser pressionado, e uma lista de tarefas renderizada usando o componente FlatList. Cada item da lista exibe a tarefa e um botão "Remover" que permite ao usuário excluir a tarefa correspondente.

Os estilos são definidos usando StyleSheet, proporcionando uma aparência organizada à aplicação. Os principais estilos incluem um contêiner que ocupa toda a tela com preenchimento e margem superior, estilos para o campo de entrada — incluindo altura, borda e preenchimento — e estilos para cada tarefa na lista, organizando os itens em uma linha com espaço entre o texto da tarefa e o botão de remoção.

Em resumo, este código exemplifica como criar uma aplicação básica de lista de tarefas em React Native, utilizando conceitos fundamentais como gerenciamento de estado, manipulação de eventos e renderização de listas. A lógica do aplicativo é clara e oferece uma experiência interativa ao usuário, permitindo adicionar

e remover tarefas facilmente. A estrutura modular do código facilita a manutenção e a adição de novas funcionalidades no futuro, tornando-se um bom exemplo para desenvolvedores iniciantes que desejam aprender a construir aplicativos móveis interativos.

```
import React, { useState } from 'react';
import {View,TextInput,Button,FlatList,Text,StyleSheet }
                                          from 'react-native';

const TodoList = () => {
  const [task, setTask] = useState('');
  const [tasks, setTasks] = useState([]);

  const addTask = () => {
    if (task) {
      setTasks([...tasks, task]);
      setTask('');
    }
  };

  const removeTask = (index) => {
    const newTasks = tasks.filter((_, i) => i !== index);
    setTasks(newTasks);
  };

  return (
    <View style={styles.container}>
      <TextInput
        style={styles.input}
        placeholder="Adicione uma tarefa"
        value={task}
        onChangeText={setTask}
      />
      <Button title="Adicionar" onPress={addTask} />
      <FlatList
        data={tasks}
        renderItem={({ item, index }) => (
          <View style={styles.task}>
            <Text>{item}</Text>
            <Button title="Remover" onPress={()=>removeTask(index)}/>
```

```
          </View>
        )}
        keyExtractor={(item, index) => index.toString()}
      />
    </View>
  );
};

const styles = StyleSheet.create({
  container: { flex: 1, padding: 20, marginTop: 50, },
  input: { height: 40, borderColor: '#ccc', borderWidth: 1,
                  marginBottom: 10, paddingHorizontal: 10 },
  task: { flexDirection: 'row', justifyContent: 'space-between',
                  marginBottom: 10 },
});

export default TodoList;
```