Vitor Amadeu Souza

Introdução ao

React Native

No ambiente Expo Snack

Parte XXV

Novembro 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 deste exemplo é um aplicativo de gerenciamento de tarefas utilizando React Native que permite ao usuário adicionar, excluir e marcar tarefas como concluídas, além de selecionar uma data para cada tarefa usando um calendário.

O aplicativo mantém um estado para as tarefas, cada uma com informações como nome, data limite, cor, e status de conclusão. As tarefas são exibidas em uma lista utilizando o componente FlatList. A cada tarefa, o usuário pode marcar como concluída ou não, e também excluir a tarefa.

O estado do aplicativo é controlado por hooks. O useState é usado para gerenciar o nome da nova tarefa, a data selecionada, e a visibilidade do calendário. A cada tarefa adicionada, um novo item é inserido na lista de tarefas, e o estado do calendário é atualizado com a data escolhida.

O componente Calendar da biblioteca react-native-calendars é utilizado para exibir um calendário onde o usuário pode selecionar uma data. O formato da data é controlado para ser compatível com o formato esperado pelas tarefas. Quando o usuário pressiona uma data no calendário, essa data é salva como a data limite da tarefa.

O estilo do aplicativo é gerido através do StyleSheet, que aplica um design simples e funcional com bordas, cores e padding em elementos como o campo de entrada de texto, o botão de adicionar tarefa, e os itens da lista de tarefas.

Cada tarefa tem um botão de exclusão que, ao ser pressionado, remove a tarefa da lista, além de um texto que permite alternar entre os estados "Concluída" e "Não concluída", com cores específicas para cada estado.

O código oferece uma funcionalidade completa de gerenciamento de tarefas, incluindo a adição, visualização, e exclusão de tarefas com um calendário interativo para selecionar datas, proporcionando uma experiência prática e intuitiva para o usuário.

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

```
import { Calendar } from 'react-native-calendars';
import { Colors } from 'react-native-paper';

const App = () => {
  const [tasks, setTasks] = useState([
    { id: 1, name: 'Tarefa 1', deadline: '2024-11-15', color: 'blue',
completed: false },
    { id: 2, name: 'Tarefa 2', deadline: '2024-11-20', color: 'red',
completed: false },
    { id: 3, name: 'Tarefa 3', deadline: '2024-11-25', color: 'green',
completed: true },
  ]);

  const [newTaskName, setNewTaskName] = useState('');
  const [selectedDate, setSelectedDate] = useState(new
Date().toISOString().split('T')[0]);
  const [showCalendar, setShowCalendar] = useState(false);

  const handleAddTask = () => {
    if (newTaskName.trim() !== '') {
      setTasks([
        ...tasks,
        {
          id: tasks.length + 1,
          name: newTaskName,
          deadline: selectedDate,
          color: 'blue',
          completed: false,
        },
      ]);
      setNewTaskName('');
      setSelectedDate(new Date().toISOString().split('T')[0]);
    }
  };

  const handleDelete = (id) => {
    setTasks(tasks.filter((task) => task.id !== id));
  };

  const handleToggleCompleted = (id) => {
    setTasks(
      tasks.map((task) =>
        task.id === id ? { ...task, completed: !task.completed } : task
      )
    );
  };

  return (
    <ScrollView>
```

```
      <View style={styles.container}>
        <Text style={styles.title}>Tarefas</Text>
        <TextInput
          style={styles.input}
          placeholder="Nome da nova tarefa"
          value={newTaskName}
          onChangeText={setNewTaskName}
        />
        <TouchableOpacity onPress={() => setShowCalendar(!showCalendar)}>
          <Text style={styles.selectDate}>Selecionar Data:
{selectedDate}</Text>
        </TouchableOpacity>

        {showCalendar && (
          <Calendar
            current={selectedDate}
            onDayPress={(day) => {
              setSelectedDate(day.dateString);
              setShowCalendar(false);
            }}
            markedDates={{
              [selectedDate]: {
                selected: true,
                selectedColor: Colors.blue500,
              },
            }}
          />
        )}

        <TouchableOpacity onPress={handleAddTask}>
          <Text style={styles.addTask}>Adicionar Tarefa</Text>
        </TouchableOpacity>
        <FlatList
          data={tasks}
          renderItem={({ item }) => (
            <View style={[styles.task, { borderColor: item.color }]}>
              <Text style={styles.taskText}>{item.name}</Text>
              <Text style={styles.deadline}>{item.deadline}</Text>
              <TouchableOpacity onPress={() =>
handleToggleCompleted(item.id)}>
                <Text style={item.completed ? styles.completed :
styles.notCompleted}>
                  {item.completed ? 'Concluída' : 'Não concluída'}
                </Text>
              </TouchableOpacity>
              <TouchableOpacity onPress={() => handleDelete(item.id)}>
                <Text style={styles.delete}>Excluir</Text>
              </TouchableOpacity>
            </View>
```