

Vitor Amadeu Souza

Introdução ao

React Native

No ambiente Expo Snack

Parte II

© 2024 by Cerne Tecnologia e Treinamento Ltda.

© 2024 by Vitor Amadeu Souza

Nenhuma parte desta publicação poderá ser reproduzida sem autorização prévia e escrita de **Cerne Tecnologia e Treinamento Ltda.** Este livro publica nomes comerciais e marcas registradas de produtos pertencentes a diversas companhias. O editor utiliza as marcas somente para fins editoriais e em benefício dos proprietários das marcas, sem nenhuma intenção de atingir seus direitos.

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.



Sumário

Capítulo I – Metodologia de desenvolvimento.....	6
1. Introdução.....	6
Capítulo II – Programação em React Native.....	7
1. Componente Image.....	7
2. Componente TextInput.....	9
4. Componente TouchableHighlight.....	15
5. Componente ScrollView.....	18

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. Componente Image

Esta aplicação usa o componente Image para carregar na tela uma imagem disponível na internet. Para isso, crie um novo *Snack* e edite o arquivo App.js com o código-fonte apresentado a seguir.

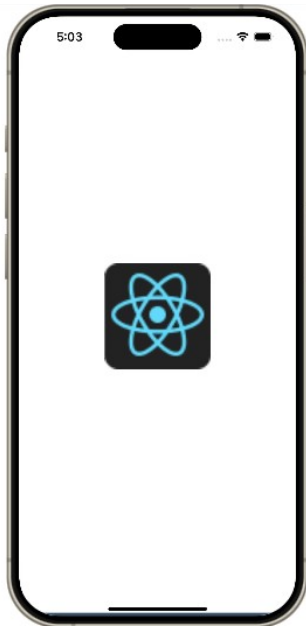
```
import React from 'react';
import { View, Image, StyleSheet } from 'react-native';

const ExemploImage = () => (
  <View style={styles.container}>
    <Image
      source={{ uri: 'https://reactnative.dev/img/tiny_logo.png' }}
      style={styles.image}
    />
  </View>
);

const styles = StyleSheet.create({
  container: {
    flex: 1, // Faz o View ocupar toda a tela
    justifyContent: 'center', // Centraliza verticalmente
    alignItems: 'center', // Centraliza horizontalmente
  },
  image: {
    width: 150,
    height: 150,
  },
});
```

O código apresentado é um componente funcional em React Native chamado `Exemplolmage`, que tem como objetivo exibir uma imagem na tela. Ele começa com a importação das bibliotecas necessárias, como `React` e componentes do `react-native`, incluindo `View`, `Image` e `StyleSheet`. Dentro do componente, uma `View` é criada para servir como contêiner da imagem. A imagem é carregada a partir de uma URL específica, que, neste caso, é `https://reactnative.dev/img/tiny_logo.png`, utilizando a propriedade `source` do componente `Image`.

Os estilos são definidos através do `StyleSheet.create`, onde duas classes são especificadas. A primeira, chamada `container`, faz com que a `View` ocupe toda a tela e centralize seu conteúdo tanto vertical quanto horizontalmente. A segunda classe, chamada `image`, define as dimensões da imagem, que são de 150 pixels de largura e 150 pixels de altura. Quando o componente é renderizado, ele exibe a imagem centralizada na tela com as dimensões especificadas, resultando em uma interface simples e visualmente agradável.



2. Componente TextInput

Este exemplo usa um componente `TextInput` como elemento de entrada e reflete em outro componente `Text` tudo que nele é digitado.

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

const ExemploTextInput = () => {
  const [value, setValue] = useState('');

  return (
    <View style={{ padding: 20, justifyContent: 'center', flex: 1 }}>
      <TextInput
        style={{
```

```

        height: 40,
        borderColor: 'gray',
        borderWidth: 1,
        paddingHorizontal: 8,
        marginBottom: 20,
        borderRadius: 5,
      }}
      placeholder="Digite algo aqui"
      onChangeText={text => setValue(text)}
      value={value}
    />
    <Text>Valor digitado: {value}</Text>
  </View>
);
};

export default ExemploTextInput;

```

Esse código é um exemplo de um componente React Native chamado `ExemploTextInput`, que utiliza o hook `useState` para gerenciar o estado do texto inserido pelo usuário. O componente principal é uma `View` que contém um campo de entrada de texto (`TextInput`) e um elemento `Text` que exibe o valor digitado.

No `TextInput`, o usuário pode digitar qualquer texto, e o valor digitado é armazenado na variável de estado `value`, inicializada como uma string vazia. O método `onChangeText` captura qualquer alteração no campo de texto e atualiza o estado com o novo valor, utilizando a função `setValue`.