

```

import React, {useState} from "react";
import { Text, View, TextInput, StyleSheet } from "react-native";

/*  TEXTINPUT:

    O TextInput é para o React o que o input é para o HTML.
    Com ele podemos adicionar um campo para recebimento de valores textuais.

    Temos que ficar atentos pois o TextInput é um Componente controlado, por isso para que possamos realmente receber valores e gerar
    alterações reais, temos que usar Controle de Estados.

    Veja como usar...

*/

//Abaixo temos um input que ao receber um valor irá atualizar a frase que há acima dele, chamamos isso de Componente Controlado, pois
ele gera uma mudança na interface gráfica...
export default () => {

    const [nome, setNome] = useState('')

    return (
        <View>
            <Text style={styles.texto}>{nome}</Text>
            {/* Aqui temos um exemplo de TextInput na prática... */}
            <TextInput style={styles.input}
                //Podemos usar um placeholder para gerar em texto com dica...
                placeholder="Digite um texto e veja o texto de cima mudar..."
                //No value iremos receber o valor que será colocado no input, note que a variável colocada aqui é uma variável gerada
                usando useState, portanto, a medida que formos digitando ou apagando a variável tem o seu valor atualizado.
                value={nome}
                //Essa propriedade é essencial para pegar o valor e fazer a mudança de estado, note que ela recebe como parâmetro uma
                callback, onde chamamos o set do useState...
                onChangeText={palavra => setNome(palavra)}
            />
        </View>
    )

```

```
}

const styles = StyleSheet.create({
  texto: {
    color: 'navy',
    fontSize: 32,
  },
  input: {
    borderWidth: 1,
    borderColor: 'navy',
    borderStyle: 'solid',
    fontSize: 24,
    margin: 10,
  }
})
```

ARQUIVO DO APP.JS...

```
import React from 'react'
import ThisIsTextInput from './src/componentes/ThisIsTextInput'

export default () => {
  return (
    /* Coloque seus componentes abaixo...*/
    <ThisIsTextInput />
  )
}
```

RESULTADO NO CELULAR ANTES...

The image shows a development environment with a mobile app preview on the left and the source code in Visual Studio Code on the right.

Mobile App Preview: The app is running on a virtual Android device. It displays a text input field with the placeholder text "Digite um texto e veja o texto de cima".

Visual Studio Code: The editor is open to the file `index.js` in the `src > componentes > ThisIsTextInput` directory. The code defines a React component with the following structure:

```
28 ..... atualizado.  
29 ..... value={nome}  
30 ..... //Essa propriedade é essencial para pegar o valor e fazer a mudança de  
31 ..... estado, note que ela recebe como parâmetro uma callback, onde chamamos o  
32 ..... set do useState...  
33 ..... onChangeText={palavra => setNome(palavra)}  
34 ..... />  
35 ..... </View>  
36 ..... )  
37 ..... }  
38 .....  
39 ..... const styles = StyleSheet.create({  
40 .....   texto: {  
41 .....     color: 'navy',  
42 .....     fontSize: 32,  
43 .....   },  
44 .....   input: {  
45 .....     borderWidth: 1,  
46 .....     borderColor: 'navy',  
47 .....     borderStyle: 'solid',  
48 .....     fontSize: 24,  
49 .....     margin: 10,  
50 .....   },  
51 ..... })  
52 ..... }
```

The **TERMINAL** pane at the bottom shows the following output:

```
BUILD SUCCESSFUL in 4m 4s  
38 actionable tasks: 2 executed, 36 up-to-date  
info Connecting to the development server...  
info Starting the app on "emulator-5554"...  
Starting: Intent { cmp=com.componentes/.MainActivity }  
[]
```

The status bar at the bottom indicates the document is in Portuguese (Brazil) and contains 285 words.

RESULTADO NO CELULAR DEPOIS...

The image shows a mobile emulator on the left and the Visual Studio Code editor on the right. The emulator displays a text input field with the text "Valores digitados depois ;P". The Visual Studio Code editor shows the file explorer on the left with the project structure for "CURSO-DE-REACT". The main editor area displays the code for "index.js" in the "styles" section, which defines the styling for the text input. The terminal at the bottom shows the build output, indicating a successful build and the start of the application on the emulator.

EXPLORADOR

- CURSO-DE-REACT
 - ThisIsButton
 - ThisIsFragment
 - ThisIsPlataform
 - ThisIsSafeAreaView
 - ThisIsTextInput
 - JS index.js
 - estilos
 - exercicios
 - fundamentos
 - truques-na-manga
 - .buckconfig
 - .eslintrc.js
 - .flowconfig
 - .gitignore
 - .node-version
 - .prettierrc.js
 - .ruby-version
 - .watchmanconfig
 - App.js
 - app.json
 - babel.config.js
 - Gemfile
 - index.js
 - metro.config.js
 - package.json
 - yarn.lock
 - consulta-rapida

index.js

```
atualizado.  
value={nome}  
//Essa propriedade é essencial para pegar o valor e fazer a mudança de  
estado, note que ela recebe como parâmetro uma callback, onde chamamos o  
set do useState...  
onChangeText={palavra => setNome(palavra)}  
/>  
</View>  
)  
}  
const styles = StyleSheet.create({  
  texto: {  
    color: 'navy',  
    fontSize: 32,  
  },  
  input: {  
    borderWidth: 1,  
    borderColor: 'navy',  
    borderStyle: 'solid',  
    fontSize: 24,  
    margin: 10,  
  },  
})
```

PROBLEMAS **SAÍDA** **CONSOLE DE DEPURAÇÃO** **TERMINAL**

BUILD SUCCESSFUL in 4m 4s
38 actionable tasks: 2 executed, 36 up-to-date
info Connecting to the development server...
info Starting the app on "emulator-5554"...
Starting: Intent { cmp=com.componentes/.MainActivity }

PAGE 4 OF 4 285 WORDS PORTUGUESE (BRAZIL)

12:37 10/12/2022