

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

```
/* COMPONENTE CONTROLADO E COMPONENTE DESCONTROLADO:
```

No React é muito importante entender a diferença entre um componente controlado e um componente descontrolado.

O QUE SÃO COMPONENTES CONTROLADOS?

Primeiramente é preciso entender de quê controle estamos falando.

Esse controle é a capacidade que um componente tem de mudar a sua interface gráfica dependendo da atualização dos seus estados de um componente.

Quando um componente é capaz de receber parâmetros e atualizá-los, gerando uma mudança na interface gráfica, chamamos esse componente de Controlado.

O QUE SÃO COMPONENTES DESCONTROLADOS?

É exatamente o oposto dos Componentes Controlados, são componentes que mesmo recebendo parâmetros, não são capazes de mudar a interface gráfica. Eles permanecem estáticos.

Veja um exemplo logo abaixo...

```
*/
```

```
//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('Nome qualquer')
```

```
  return (
```

```
    <View>
```

```
      <Text style={styles.texto}>{nome}</Text>
```

```
      <TextInput style={styles.input}
```

```

        placeholder="Digite um texto e veja o texto de cima mudar..."
        value={nome}
        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 Visual Studio Code and an Android emulator. The emulator displays a simple app with a text input field. The code in the background is as follows:

```
1 import React from 'react'
2 import ThisIsTextInput from '../src/componentes/ThisIsTextInput'
3
4 export default () => {
5   return (
6     /* Coloque seus componentes abaixo...*/
7     <ThisIsTextInput />
8   )
9 }
10
```

The terminal output shows the following build process:

```
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 }
```

RESULTADO NO CELULAR DEPOIS...

The image shows a development environment with Visual Studio Code and an Android emulator. The emulator displays the text "Coloquei palavras aleatórias" in a text input field. The Visual Studio Code editor shows the file explorer with the "CURSO-DE-REACT" project structure. The "index.js" file is open, showing the component structure and the "ThisIsTextInput" component. The terminal at the bottom shows the build process and the app starting on the emulator.

Visual Studio Code interface details:

- File Explorer: **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
 - JS App.js
 - app.json
 - babel.config.js
 - Gemfile
 - JS index.js
 - metro.config.js
 - package.json
 - yarn.lock
 - > consulta-rapida

Terminal 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 }
```