

Com amor, **styled-components**

& typescript w/ react-native

hroad



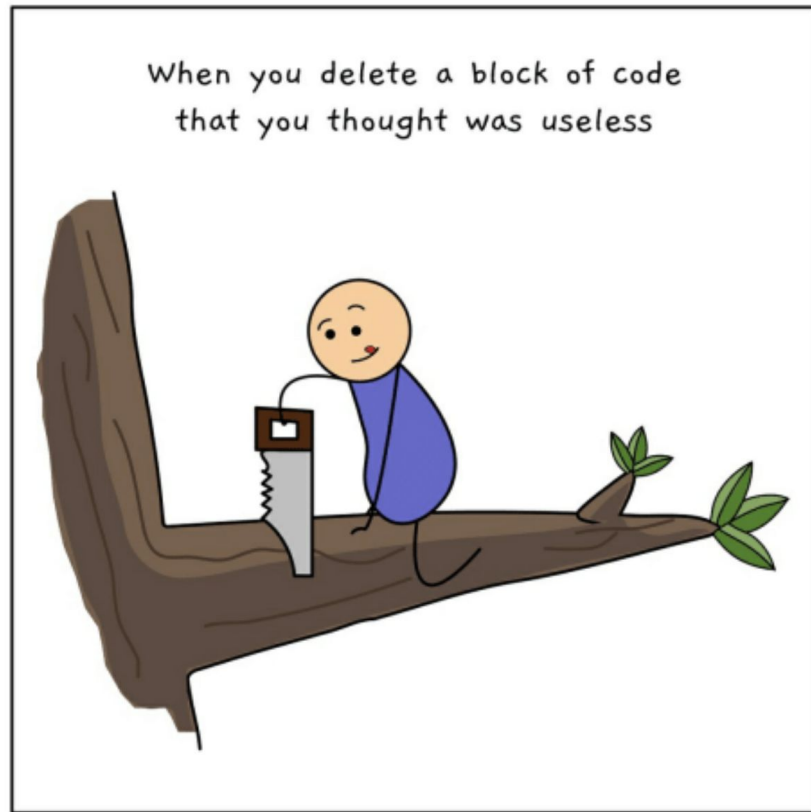
A dedicação é singular pra mim.

O que veremos

- ❑ Contexto
- ❑ Design
- ❑ Implementação
- ❑ Criando um componente com styled-components
- ❑ Dicas Extras
- ❑ Um pouco sobre o nosso desenvolvimento

Dois pontos importantes

Manutenibilidade
Softwares Escaláveis



tela a

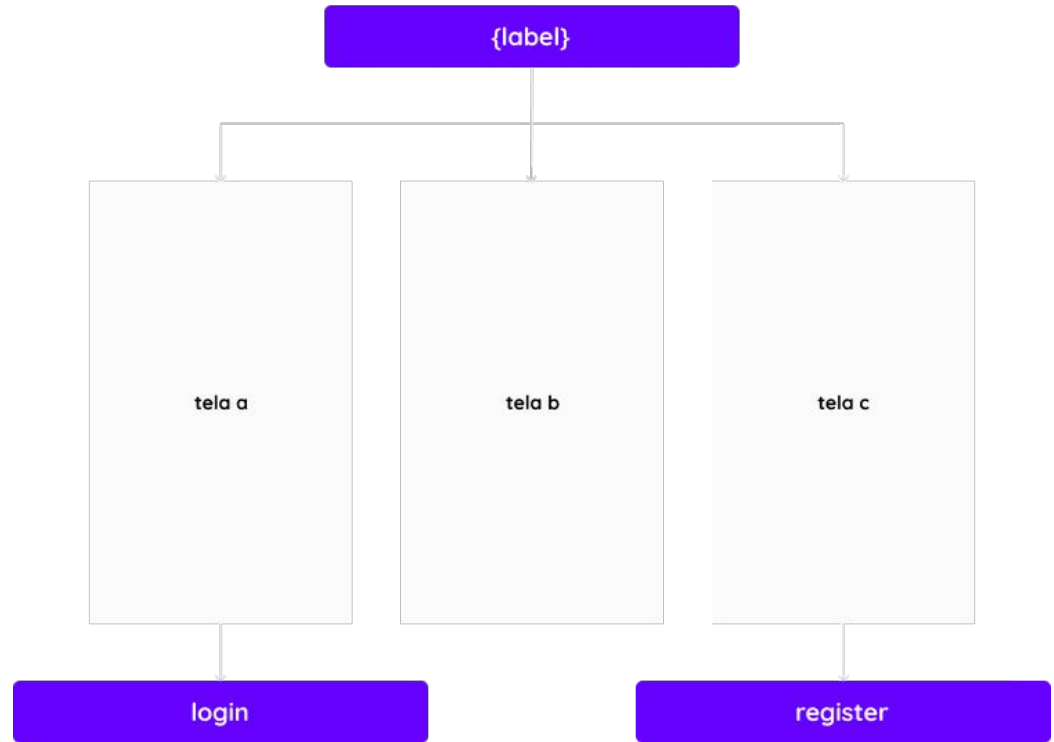
login

tela b

register

exemplo de uma aplicação

Componente FlatButton



exemplo componentes



```
import React from 'react';

import styled from 'styled-components/native';

import { Props, PropsStyled } from './interface';

const Container = styled.TouchableOpacity<PropsStyled>`
  background-color: ${props => (props.disabled ? 'green' : 'blue')};
`;

const Label = styled.Text``;

const OutlineButton = ({ text, disabled }: Props) => {
  return (
    <Container disabled={disabled}>
      <Label>{text}</Label>
    </Container>
  );
};

export default OutlineButton;
```

```
export interface Props {
  text: string;
  disabled?: boolean;
}

export interface PropsStyled {
  disabled?: boolean;
}
```

criando um componente


```
const App = () => {  
  return (  
    <SafeAreaView>  
      <OutlineButton text="Hello ~ Flat" disabled={true} />  
    </SafeAreaView>  
  )  
}
```

Hello ~ Outline

utilizando este componente

```
const App = () => {  
  return (  
    <SafeAreaView>  
      <OutlineButton text="Hello ~ Flat" disabled={false} />  
    </SafeAreaView>  
  )  
}
```



Hello ~ Outline

utilizando este componente

resultado e facilidade de manutenção de uma aplicação componentizada.

Organizando

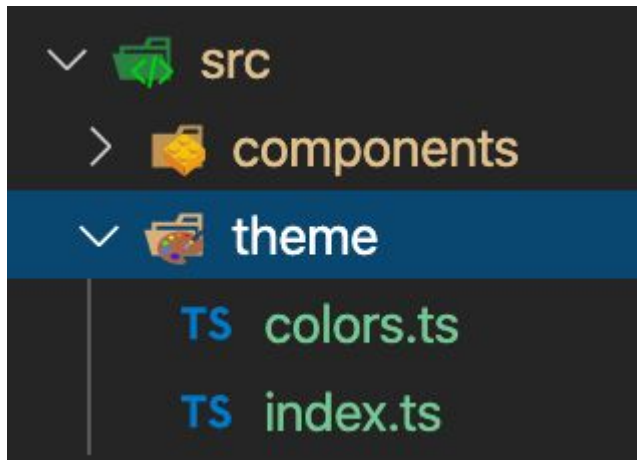
- A. Instalar a dependência;
- B. `styled.[nome-react-native] ~ styled.Text`;`
- C. `css-in-js;`
- D. Sem side-effects;
- E. Sem nomes de classes;

Dicas++

Declarando um tema;

Elegância nos componentes;

ThemeProvider



Você também pode utilizar para fontes e unidades, por exemplo.



```
export default {  
  purple: '#6600ff',  
};
```

src/theme/colors.ts



```
import { DefaultTheme } from 'styled-components';

import colors from './colors';

declare module 'styled-components' {
  export interface DefaultTheme {
    colors: typeof colors;
  }
}

export const theme: DefaultTheme = {
  colors,
};
```

src/theme/index.ts

incluir imagem aqui do themeprovider theme={theme}

ThemeProvider

```

const Container = styled.TouchableOpacity<PropsStyled>`
  background-color: ${props => (props.disabled ? 'green' : 'white')};
`;

const Label = styled.Text`
  color: ${({ theme }) => theme.colors};
`;

```

colors (property) DefaultTheme.colors: {
purple: string;
}

```

const Label = styled.Text`
  color: ${({ theme }) => theme.colors.purple};
`;

```

purple (property) purple: string

utilizando as cores disponibilizadas pelo tema

```

/**
  ...
  Example: <TextButton text="Hello World" disabled={true} />
  ...
*/
const TextButton = ({ text, disabled }: Props) => {
  return (
    <Container disabled={disabled}>
      <Label>{text}</Label>
    </Container>
  );
};

```

```

const TextButton = ({ text, disabled }: Props) => {
  return (
    <Container disabled={disabled}>
      <Label>{text}</Label>
    </Container>
  );
};

(alias) const TextButton: ({ text, disabled }: Props) => JSX.Element
import TextButton

Example: <TextButton text="Hello World" disabled={true} />

Property 'text' is missing in type '{}' but required in type 'Props'. ts(2741)
interface ts(2, 3): 'text' is declared here.
<TextButton />

```

E um pouco de elegância

Um pouco sobre nosso desenvolvimento

Tempo inicial maior na criação dos componentes;

Desafio de algo novo;

Robustez;

Genéricos;

Reutilização;

Escalabilidade;