Nosso Desafio
Design & Implementação
Nosso Desenvolvimento
Dicas Extras ~ Compactadas

Olá, eu me chamo hroad (apenas brincando e sem pesquisar no google, por favor) e eu vim contar um pouco sobre nosso desenvolvimento e nossos desafios.

Quando nós começamos a desenvolver o app, eu acho que um dos nossos primeiros desafios era desenvolver uma aplicação escalável, que sua manutenção ou necessidade de alteração tivesse um impacto “pequeno” e que não tivesse side-effects.

Pensando na problemática de ter telas com componentes iguais, como evitarmos repetição de código, evitar ctrl+c, ctrl+v
Como facilitar a mudança, de cores, fontes, unidades
Se fosse em todas as telas, o desenvolvimento iria ser demorado
E não garantiria a integridade
Além disso, nós tínhamos componentes que seguiam toda uma padronização de informações e/ou estilos
Pensando nesse contexto...

Uma das primeiras coisas que fizemos/pensamos fora como como componentizar nossa aplicação
Olhando este componente de FlatButton, por exemplo, ele pode receber um texto e uma vez que eu precise realizar uma alteração de cor, por exemplo, eu altero ele neste ponto e é refletido em todo meu código.

Bom, e o que escolhemos para desenvolver tudo isto? Fora styled-components e typescript

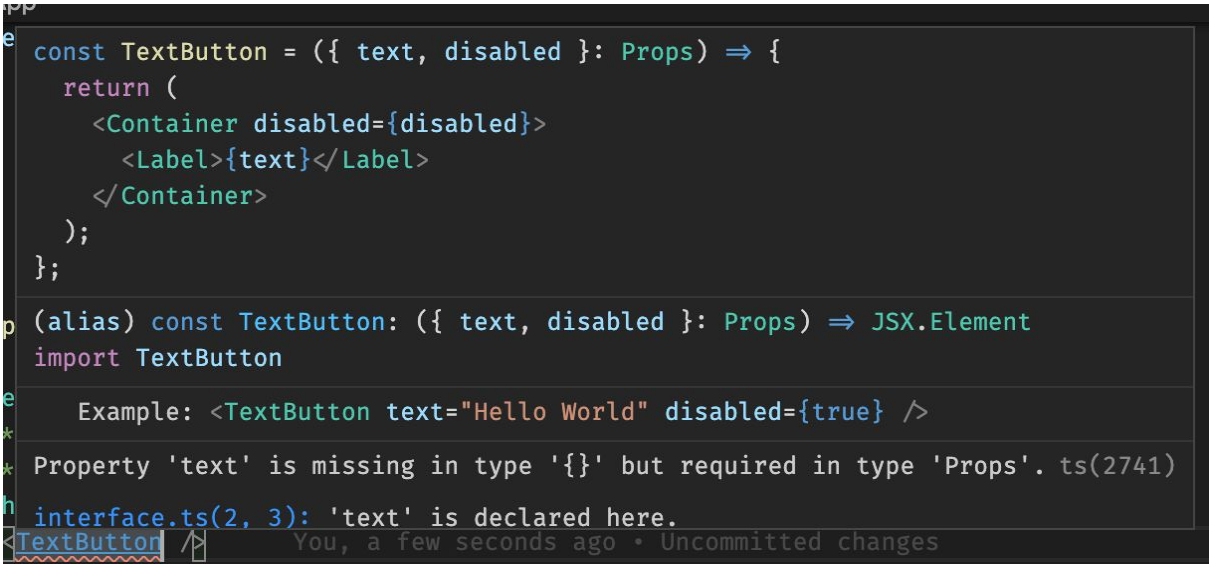
E contando um pouco sobre a minha experiência e um pouco dos meninos,
Tempo de construção no início maior (lembrar da montanha russa)
Tempo de construção da tela em si (menor), sem gerenciamento de estado e/ou api
facilidade de alteração/manutenção
As vezes voltávamos para ajustar props, estilo no início, e depois isso também diminuía

Tá bom, mas como a gente coloca tudo isso em prática?
Vou mostrar primeiro um exemplo de como criar um componente com styled-components e depois um exemplo mais completo do que podemos fazer.
E dois tópicos extras. Que dê tempo! Bora nessa.

Exemplo de styled.Text``
Exemplo Completo

Uau, está ficando um pouco maior meus componentes, nossa estrutura. Como eu sei o que cada componente irá receber? Quais props ele precisa receber? Você pode criar uma anotação em cima do componente a fim de facilitar a visualização e verificar tudo o que ele precisa para ser utilizado.

Incrível, não?



ThemeProvider

Pensando na parte de design também, design tokens, utilização e manutenção de cores, fontes, unidades, etc
Nós conseguimos dispor o tema para todos os componentes através da context api e dispor toda nossa informação

E o typescript além de outros fatores, tem a tipagem e o autocomplete. Como a gente acaba

definindo um tema, a gente define toda a sua interface, tudo que ele irá dispor de informação para os componentes.
Tudo que estará disponível.

Para que a apresentação não fique extensa, eu vou deixar no nosso próximo e penúltimo slide, uma aplicação completa com tudo o que falamos aqui.