

Otimizando código

Transcrição

[0:00] Uma última coisa que a gente pode fazer nesse código que a gente criou da funcionalidade de login é o quê? Se a gente for analisar, a gente chama três vezes o método. Ou melhor, a gente chama duas vezes o `pegarInputUserName` e `pegarInputPassword`.

[0:19] Se a gente for no page object aqui, a gente utiliza quase as mesmas informações, assim, não muda nada.

[0:27] Então, o que a gente pode fazer? A gente pode otimizar o nosso código.

[0:31] Que seria o quê? A gente cria um genérico `pegarInput` e pode colocar dois parâmetros nele, que no caso se a gente for verificar aqui, o que ele muda?

[0:42] O `formControlName` nele a gente pode trocar qual que é o nome do `formControl`, e também podemos trocar o `sendKeys` dele. Então, precisamos enviar dois parâmetros aqui.

[0:56] Que é o quê? O `formControlName` e o valor.

[1:07] Certo? Então vou dar um `return`, `element by css input formControl`.

[1:19] Uma coisa que a gente precisa trocar aqui. Na verdade, a gente não pode colocar aspas simples. Tem que colocar crase que a gente usa `template string`.

[1:29] Ok, `input formControlName` igual.

[1:39] A gente coloca o cifrão, abre chaves e passa a variável para dentro dele.

[1:46] Isso são coisas do ECMAScript. Acredito que o ECMAScript 6, que foi implementado as informações, por exemplo, colocar `template string`, para que facilite o nosso código, para a gente não fazer aquele binding com um maizinho e ficar criando mais informações, que fica gerando código demais.

[2:06] Por exemplo, antes a gente precisava fazer... não conseguia fazer assim. Vou só comentar aqui. Vou dar um `Ctrl C`, `Ctrl V`, vou descomentar.

[2:18] Para comentar, você dá `Ctrl` e a tecla do ponto e vírgula.

[2:24] Então, antes, como é que fazia? Eu criava uma aspas simples aqui. Eu não conseguia colocar variável dentro de uma `string`. Tinha que fazer o quê?

[2:36] Fechar aspas aqui, dar um `mais`, abre outra aspas simples, colocar o nome da variável, ou melhor, colocar o nome da variável aqui não precisa de aspas simples; aí eu faço um `mais` para fazer um binding de outra variável, de outra `string`. E por assim vai.

[2:54] Então, com o ECMAScript 6 facilitou muito com a utilização de `template string`.

[3:00] Essas coisas assim você pode ver no nosso curso de ECMAScript 6, certo, que pode facilitar muito a sua vida na hora de você criar um código que utiliza Javascript.

[3:15] Agora, eu vou enviar um `sendKeys` e colocar o valor.

[3:19] Agora, quem vai ficar dependente de enviar as informações?

[3:25] O próprio `signIn` e `e2e`. Voltei para ele mesmo. Vou chamar agora não mais `username`, vou pegar só `input` e passar dois parâmetros.

[3:36] No caso, o primeiro é uma `string`, que é `username` e o segundo parâmetro é uma `string`, que é Flávio.

[3:47] Aqui embaixo, a mesma coisa.

[3:51] É uma variável, ou melhor, vou chamar o `getElement`.

[3:56] E vou passar o `password` e enviar a senha 123.

[4:04] Esse código aqui, voltando para o `page object`, eu posso deletar ou pegar `input password`, pegar `input username` e posso deletar.

[4:13] Otimizei o nosso código aqui de uma maneira muito mais simples.

[4:17] Poderia fazer outras coisas aqui. Ao invés de digitar `expect` e gerar esses dois códigos, eu poderia fazer um `forit`, declarar esses dois em um objeto, mas eu acho que ficaria muito mais trabalhoso. Assim, `Ctrl C Ctrl V` está muito mais fácil.

[4:33] Mas vamos só testar para ver se não deu nenhum erro.

[4:36] Claramente a gente otimizou o código, voltando aqui para o código, otimizamos o código.

[4:42] Mas é sempre bom a gente testar para ver se não quebramos nada, se realmente tudo o que fizemos está tudo certo.

[4:49] Voltei para o nosso `prompt` e vamos ver se está executando como estava executando como antes.