

Utilizando element e By

Transcrição

[0:00] Agora a gente precisa testar a nossa funcionalidade de login.

[0:04] Essa funcionalidade de login simplesmente vai contemplar o username, no campo de input de username, o outro campo de input de password, de senha, e um botão de login.

[0:22] Mas como é que a gente vai fazer isso?

[0:23] A gente precisa testar como.

[0:25] Vamos percorrer então os nossos elementos do DOM, capturar esses três elementos, esses dois campos de input e esse campo de botão, e enviar informações para eles dizendo esse cara está aceitando isso, existe mesmo esse campo na tela e ele aceita esse tipo de string, esse tipo de parâmetro, enfim, a gente vai fazer esse teste.

[0:49] Mas como faremos isso?

[0:51] Vamos aqui na nossa API, eu já deixei aberta ela aqui, a API do Protractor, e ele tem um camarada aqui chamado element, esse locator aqui.

[1:03] O que ele faz?

[1:04] Toda a vez que eu dou um element alguma coisa, element procure por alguma classe, element procure por algum edi, ele vai fazer o quê?

[1:14] Esse element vai nos trazer todos os elementos que estão no DOM de acordo com a página que a gente estiver navegando, que estiver testando, e ele vai dizer "toma, esses são todos os elementos que tem aqui no seu DOM nesse exato momento".

[1:35] E o que a gente precisa nesse exato momento?

[1:38] Precisa pegar ou pela classe, ou pelo FormControlName, ou pela tag name desse camarada aqui, desse camarada e desse aqui também.

[1:49] Mas como é que a gente vai fazer isso?

[1:52] Vou voltar aqui para a código nesse signin.po.ts, já vou importar esse element, que já vimos que a gente vai precisar dele para ele percorrer os elementos do nosso DOM.

[2:08] E vou fazer o quê?

[2:08] Criar um método.

[2:10] validarLogin.

[2:16] E vou dar um element.

[2:20] Só que esse element, nesse caso a gente precisa saber de alguma característica dos nossos campos.

[2:30] Se a gente voltar para o nosso projeto, ele tem dois campos aqui.

[2:36] Mas quais informações que contemplam eles, sabe?

[2:39] Não tem como saber.

[2:41] Nesse meio tempo, deixa eu só explicar que o ideal no nosso caso, como a gente está fazendo teste nós precisaríamos de uma documentação, porque não é o ideal que a gente fique procurando coisas ou no código, ou no nosso projeto para que nós criemos o nosso código.

[3:00] Entretanto, como nós não temos a documentação em si, eu quero mostrar mais para vocês a ferramenta, como ela funciona, a ferramenta Protractor, e a ferramenta Protractor faz esse interação de pegar elementos do DOM e fazer com que nós consigamos fazer esses testes, essa interação.

[3:24] Então o que a gente vai precisar fazer, já que a gente olhou no nosso projeto que está navegável e não encontrou algumas particularidades muito diferentes uma da outra?

[3:34] A gente vai fazer o que: dou um `CONTROL P` aqui no meu VS Code e vou procurar por `signin.component.html`, porque eu preciso do HTML agora.

[3:53] A princípio, eu vou querer só pegar esse input aqui primeiro.

[4:01] Que é esse `userName`, ele tem esse placeholder que está exatamente na tela de login que a gente viu, só que qual que é o problema?

[4:09] Eu tenho dois inputs.

[4:11] Eu preciso ver alguma coisa diferente entre eles.

[4:15] Logo de cara, a gente já consegue ver que o `formControlName` é diferente.

[4:21] Que o tipo dele também é diferente.

[4:22] Esse daqui, por mais que não esteja explícito tem um tipo diferente.

[4:26] Esse daqui é tipo `text`, e esse aqui é tipo `password`.

[4:31] A classe é a mesma, e o placeholder é diferente.

[4:38] Podemos fazer então de algumas maneiras ele.

[4:41] Que é o que?

[4:43] Esse element vai fazer a busca pelo que, primeiro?

[4:47] Deixa eu pensar.

[4:48] O ideal aqui, o mais fácil seria buscar pela classe.

[4:55] Sem dúvida.

[4:55] Mas o que a gente pode fazer é utilizar esse `FormControlName`.

[4:59] Então vamos voltar para o `signing.po`, e eu vou fazer o quê?

[5:03] Ele vai fazer uma busca, esse `element by.css`.

[5:10] Esse `by.css`, eu já vou importar esse `by` aqui.

[5:14] Esse `by.css`, na verdade, ele faz uma busca um pouco genérica.

[5:19] Às vezes você pode procurar pelo `class name`, sem problema, mas esse `css` também pode procurar pelo `class name`.

[5:28] Se eu digitar aqui, por exemplo, `FormControl`, que é o nome da classe que a gente tem lá, ele faria essa busca tranquilo.

[5:36] Só que qual que é o problema?

[5:36] A gente não consegue fazer pela classe porque se a gente voltar para o `signin.component.html`, esses dois têm a mesma classe.

[5:46] Então vamos fazer pelo quê?

[5:49] Eu vou digitar aqui `input`, porque ele consegue pegar pelo nome da tag dele, abrir colchetes aqui, abrir e fechar e passar algum parâmetro para ele.

[6:00] Por exemplo, se eu quisesse pegar, no caso, por qualquer parâmetro que ele tenha, eu conseguiria pelo tipo dele, pelo `place holder` dele, pelo `FormControlName` dele.

[6:12] Nesse caso, a gente vai pegar pelo `FormControlName`.

[6:16] Então `FormControlName` igual, eu abro chaves duplas aqui porque aqui fora eu já tenho uma chave simples, e vou digitar `userName`.

[6:29] `userName`.

[6:35] E nesse caso aqui, depois que eu fiz essa busca por esse elemento, vou precisar enviar alguma informação para ele dizer "esse campo `input` está recebendo informação tal para a gente seguir".

[6:47] E como é que eu faço isso?

[6:49] Utilizando o `sendKeys`, que ele vai receber alguma variável que seja, alguma `string` ou qualquer coisa que nós quisermos enviar, eu vou no caso digitar `Flavio`.

[7:07] Vou salvar aqui e vou chamar esse `validarLogin`, e vamos só fazer um teste para ver se realmente está funcionando tudo certinho até agora.

[7:15] Então eu vou dar um `it`, `if` não, não agora, fazer `login`.

[7:26] No caso, não é o `login` que a gente está testando tudo agora, mas essa funcionalidade vai ser para testar o `login`.

[7:32] Então já vou deixar o `login` e vou dizer que um `expect` eu quero que o `signin page` ponto, acho que eu digitei errado aqui, `signinPage`., eu dou um `control` espaço e ele me mostra os métodos que têm lá dentro, eu quero `validarLogin`, eu quero chamar esse `validarLogin`.

[8:04] Eu quero que ele só retorne `true`, que dê tudo certo, então não preciso colocar nenhuma informação aqui ou `toBe` `equal`, `toEqual`, `toBe`, alguma coisa assim.

[8:14] Eu só quero que ele retorne true e valide.

[8:16] Beleza, salvei.

[8:18] Vamos lá no nosso Prompt de comando, vou dar um `ng e2e` e vamos ver o que vai acontecer.

[8:29] Está subindo o nosso projeto tranquilamente.

[8:36] Beleza, vai fazer o nosso teste.

[8:41] E aparentemente deu um erro.

[8:42] Testando tela home deve fazer login, bem nesse nosso teste `fail` no `element found using locator: By(css selector, input[FormControlName="username"])`.

[8:59] Está errado.

[9:01] Vamos voltar para o nosso código, para o nosso HTML.

[9:04] `userName`, esqueci um `R` aqui.

[9:08] Vamos voltar para o nosso `page object`.

[9:10] Já sabemos que pelo menos ele capturou o nosso dado.

[9:15] Capturou aqui e ele está validando.

[9:20] Então vamos só continuar, ou melhor, vamos rodar novamente aqui no nosso Prompt `ng e2e` para ver se está rodando tudo certo, mas é bom já saber que pelo menos ele já não encontrou nenhuma tag com aquele `FormControlName`, perfeito?

[9:40] Executou o nosso código, vamos ver se agora vai dar tudo certo com o nosso teste.

[9:47] Perfeito, deve fazer login.

[9:50] Então vamos voltar para o nosso código e enviar mais informações para ele.