05

Totalizando o volume em nosso Template com a função reduce

Transcrição

Podemos fazer o código de maneira funcional, sem precisar usar a "gambiarra" feita na tag `<tfoot>` para exibirmos o total do `volume`:

```
<tfoot>
    <td colspan="3"></td>
    <td>
      ${
        (function() {

            let total = 0;
            model.negociacoes.forEach(n => total+= n.volume);
            return total;
       })()
      }
   </td>
</tfoot>
```

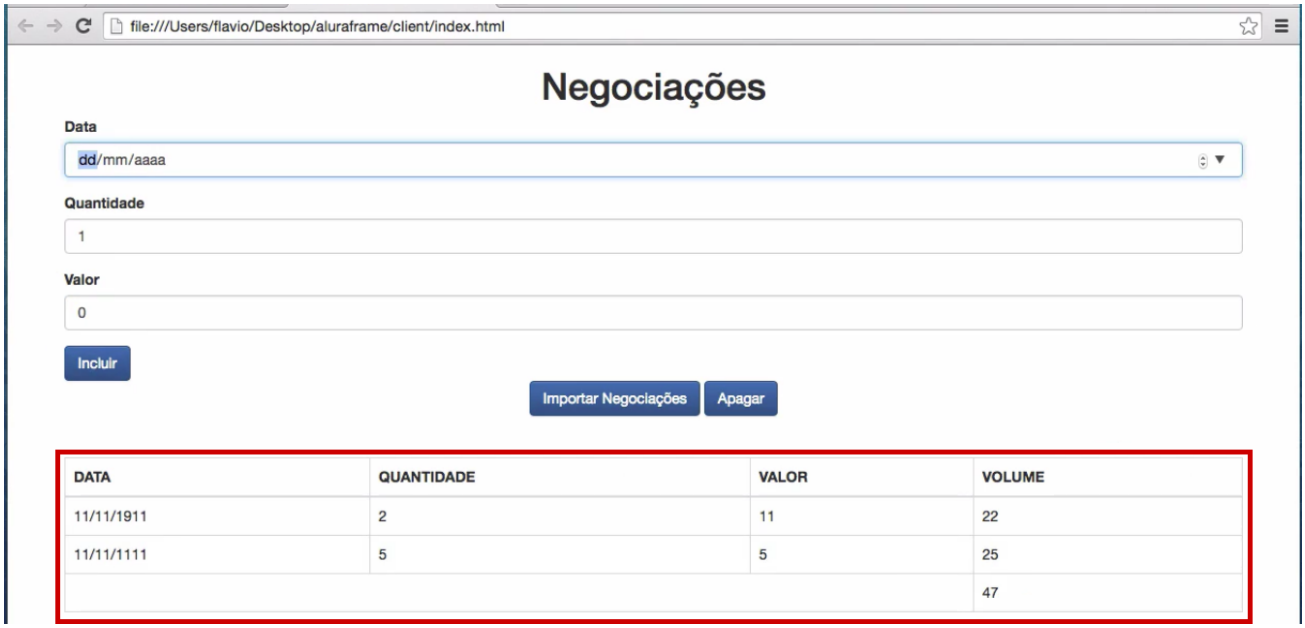
Vamos mostrar como conseguir o mesmo resultado usando o paradigma funcional e como o JavaScript *array* é bastante poderoso.

```
<tfoot>
    <td colspan="3"></td>
    <td>
        ${model.negociacoes.reduce(function(total, n) {
               return total + n.volume;
         }, 0.0)
        }
     </td>
</tfoot>
```

Observe que utilizamos a função `reduce()`, que irá processar o array e no fim, disponibiliza um único resultado. Primeiramente, não utilizaremos *arrow functions*. Optamos por passar uma função com as variáveis `total` e `n` (elementos da lista) - ambas receberam esses nomes, mas poderíamos ter definido outros. O `return` que criamos ainda não será suficiente. Qual será o valor inicial de `total`? Ele deve iniciar de `0` para conseguirmos somá-lo com `volume`. Por isso, o segundo parâmetro da função `reduce()` será a inicialização da variável `total`.

Basicamente, nós pedimos que `negociacoes` reduzisse. Em seguida, executamos a função para cada item da lista. A variável `total` começou com o valor igual a `0` e foi somado com o `volume`. Quando passamos para o segundo item da lista, este pega o valor anterior e o soma com o volume atual. No fim, a função retorna um valor único, que será o resultado de `total`.

Ao executarmos o código, veremos que ele funciona perfeitamente:

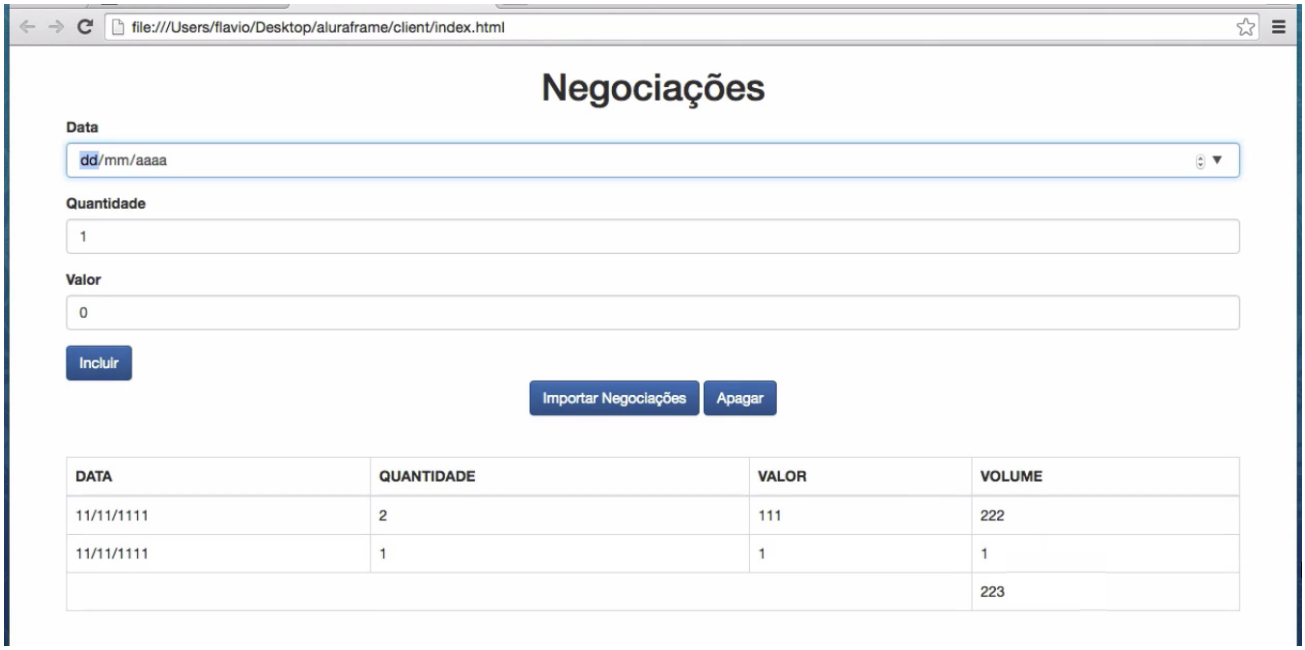


Temos o valor correto do total. Agora vamos melhorar o código, utilizando a *arrow function*. No caso, como estamos trabalhando com dois parâmetros, não podemos remover os parênteses, mas podemos eliminar o `function`. Depois de adicionarmos a flecha, podemos remover as chaves `{}`. Poderemos também remover o `return()`. Com as alterações, a tag `<tfoot>` ficará assim:

```
<tfoot>
    <td colspan="3"></td>
    <td>
        ${ model.negociacoes.reduce((total, n) => total + n.volume, 0.0)}
    </td>
<tfoot>
```

A função `reduce()` executará uma *arrow function*, que recebe como parâmetro as variáveis `total` e `n`. Cada vez que varrermos os elementos do *array*, o `total` será o que tínhamos (inicializando pelo `0`) somado ao `volume`. No fim, o `reduce()` retornará o resultado de `total`. É uma maneira funcional de lidarmos com problema para totalizar o `volume`. Como a função retornará um único valor, não foi necessário utilizar a *IIFE* para incluirmos várias instruções dentro do `forEach()`.

Vamos executar o código. Após preenchermos duas vezes o formulário, teremos o valor correto do total de `volume`.



Terminamos o template da tabela. A cada negociação incluída, a informação será exibida para o usuário com base nas informações da lista.