02

Isolando a responsabilidade de conversão de datas

Transcrição

Ao observarmos o código descobrimos que ele possui poucas linhas. Para criarmos uma data, o código deverá ser executado sempre que adicionarmos e imprimirmos uma data.

```
let data = new Date(...
    this._inputData.value
    .split('-')
    .map((item, indice) => item - indice % 2)
);

let negociacao = new Negociacao(
    data,
    this._inputQuantidade.value,
    this._inputValor.value
);

let diaMesAno = negociacao.data.getDate()
    + '/' + (negociacao.data.getMonth() + 1)
    + '/' + negociacao.data.getFullYear();

console.log(diaMesAno);
```

O nosso objetivo será isolar a responsabilidade de instanciar uma data e exibir a data num formato já conhecido numa classe. Esta será uma classe ajudante que chamaremos de `helper` e terá a responsabilidade isolada de lidar com a data.

Agora, dentro das pasta `Helpers` , criarei o arquivo `DateHelper.js` . Dentro da classe, teremos dois métodos: o primeiro método será `textoParaData()` , que receberá um texto e o converterá para data. O segundo será o `dataParaTexto` , que receberá uma data e a converterá em texto.

```
class DateHelper {

    textoParaData(texto) {

    }

    dataParaTexto(data) {

    }
}
```

Aproveitaremos o código que faz a conversão no `NegociacaoController` e adicionaremos nos métodos:

```
class DateHelper {

    textoParaData(texto) {
```

```
        return new Date(...texto.split('-').map((item,indice) => item - indice % 2));

    }

    dataParaTexto(data) {

        return data.getDate()
            + '/' + (data.getMonth() + 1)
            + '/' + data.getFullYear();
    }
}
```

Fizemos alguns pequenos ajustes, por último, inverteremos a ordem dos métodos para que eles sigam a ordem usada anteriormente.

```
class DateHelper {

    dataParaTexto(data) {

        return data.getDate()
            + '/' + (data.getMonth() + 1)
            + '/' + data.getFullYear();
    }

    textoParaData(texto) {

        return new Date(...texto.split('-').map((item,indice) => item - indice % 2));

    }
}
```

Em seguida, carregaremos este script no `index.html` .

```
<!-- ... -->
 <script src="js/app/models/Negociacao.js"></script>
 <script src="js/app/controllers/NegociacaoController.js"></script>
 <script src="js/app/helpers/DateHelper.js"></script>
 <script>
     let negociacaoController = new NegociacaoController();
 </script>
</body>
</html>
```

Faremos alterações também no `NegociacoesController.js` , apagaremos as variáveis `data` e `diaMesAno` geradas anteriormente e vamos criar uma nova `data` :

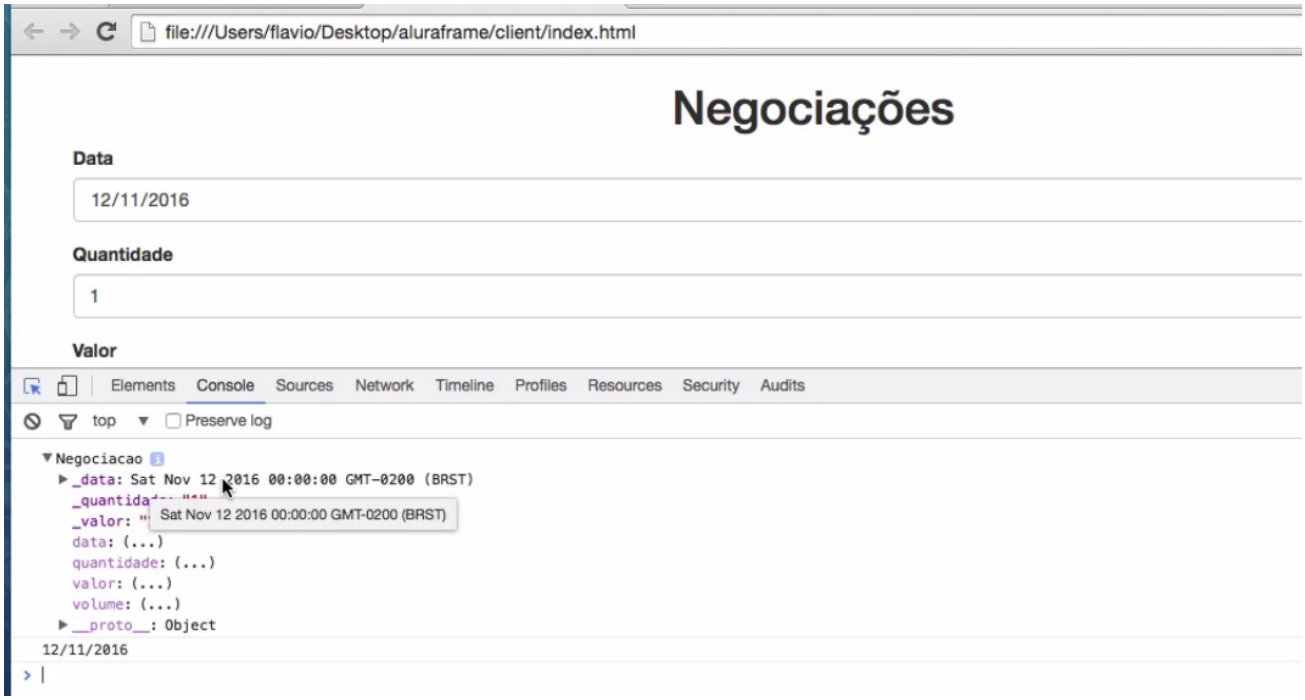
```
adiciona(event) {

        event.preventDefault();

        let data = new DateHelper().textoParaData(this._inputData.value);

        let negociacao = new Negociacao(
```

```
                data,
                this._inputQuantidade.value,
                this._inputValor.value
            );


        console.log(negociacao);

        console.log(diaMesAno);
    }
}
```

Observe que adicionamos o `DateHelper` dentro do `data` . Agora, queremos que a data da negociação seja exibida no formato `diaMesAno` , separada com barras `/` . Em vez de usarmos o `new` no `DateHelper` , vamos adicionar a variável `helper` . Em `data` , adicionaremos a `helper` que dará o texto para `data`

```
adiciona(event) {

        event.preventDefault();

        let helper = new DateHelper();
        let data = helper.textoParaData(this._inputData.value);

        let negociacao = new Negociacao(
                data,
                this._inputQuantidade.value,
                this._inputValor.value
            );


        console.log(negociacao);

        console.log(helper.dataParaTexto(negociacao.data));
    }
}
```

Agora, isolamos a responsabilidade das conversões de data para classe `Helper` . Se executarmos o código, será criada a data correta:



A data também é impressa corretamente no formato texto. A classe `Helper` cumpriu bem o seu trabalho.

Em seguida, faremos um novo ajuste. Em vez de utilizarmos a variável `data`, moveremos a seguinte linha para a `negociacao`:

```
helper.textoParaData(this._inputData.value)
```

Com a alteração, o código ficará assim:

```
adiciona(event) {

    event.preventDefault();

    let helper = new DateHelper();

    let negociacao = new Negociacao(
        helper.textoParaData(this._inputData.value),
        this._inputQuantidade.value,
        this._inputValor.value
    );
}
```

Não precisamos mais da variável `data`, usaremos diretamente o seu retorno.

Vamos testar o código no navegador, e veremos que a data `12/11/2019` aparecerá no Console.

Nós conseguimos isolar o código dentro do `Helper`. Mas será que ainda podemos melhorá-lo ainda mais? Sempre que falo isso, vocês sabem que é possível. Veremos como, mais adiante.