07

Atualizando o Item na View com JavaScript

Transcrição

Começaremos a testar o objeto de resposta, pegar o resultado e começar a atualizar a nossa página de carrinho. Voltaremos ao nosso código JavaScript, no arquivo `carrinho.js`

```
    postQuantidade(data) {
        $.ajax({
            url: '/pedido/updatequantidade',
            type: 'POST',
            contentType: 'application/json',
            data: JSON.stringify(data)
        }).done(function (response) {

                });
    }
}

var carrinho = new Carrinho();
```

Inseriremos a instrução `debugger` que fará com que o navegador Chrome pare na linha correta.

```
    postQuantidade(data) {
        $.ajax({
            url: '/pedido/updatequantidade',
            type: 'POST',
            contentType: 'application/json',
            data: JSON.stringify(data)
        }).done(function (response) {
                    debugger;

                });
    }
}

var carrinho = new Carrinho();
```

Na página de carrinho no navegador, clicaremos sobre o botão "+" para acrescentar um novo item. Assim que fizemos esse procedimento, iremos ser direcionados para a linha em que se encontra o comando `debugger` , dessa forma poderemos inspecionar o objeto `response` . Esse objeto contém `ItemPedido` , que conterá as informações do item que teve sua quantidade atualizada, e `carrinhoViewModel` , que por sua vez abriga todas as informações acerca do carrinho de compras.

Acessaremos o objeto `ItemPedido` que está dentro de `response` . Em nosso código JavaScript criaremos uma variável local que declararemos como `itemPedido` , que será igual a `response.itemPedido`

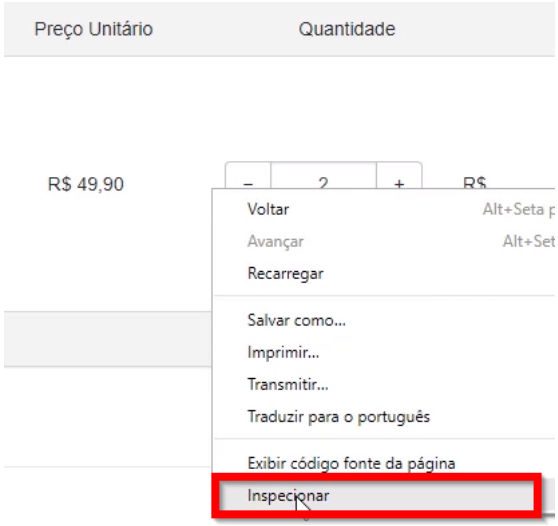
```
    postQuantidade(data) {
        $.ajax({
            url: '/pedido/updatequantidade',
            type: 'POST',
```

```
            contentType: 'application/json',
            data: JSON.stringify(data)
        }).done(function (response) {
                    let itemPedido = response.itemPedido;
                    debugger;

                });
    }
}

var carrinho = new Carrinho();
```

Dessa forma, ao voltarmos para a página de carrinho e selecionarmos o botão "+", `ItemPedido` será preenchido com as informações que precisamos para atualizar a linha de pedido. Sobre a área de "Quantidade", clicaremos com o botão direito do mouse e selecionaremos a opção "Inspecionar".



Dessa forma, veremos o código html da página. Para chegarmos exatamente na linha que corresponde ao item atualizado, procuraremos o atributo `item-id` .

```
<div class="row row-center linha-produto" item-id="22011"> ==$0
```

Obteremos a parte do objeto de resposta para chegarmos na `<div>` . Voltaremos ao código JavaScript e declararemos uma variável `linhaDoItem` que será obtida a partir do jQuery. Para fazermos uma consulta a esta biblioteca, iremos inserir `$('[item-id=' + itemPedido.id +]')` , ou seja, o seletor com uma concatenação de string com `itemPedido.id` .

```
    postQuantidade(data) {
        $.ajax({
            url: '/pedido/updatequantidade',
            type: 'POST',
            contentType: 'application/json',
            data: JSON.stringify(data)
        }).done(function (response) {
                    let itemPedido = response.itemPedido;
                    let lihaDoItem = $('[item-id=' + itemPedido.id +]')
                    debugger;

                });
    }
```

```
}

var carrinho = new Carrinho();
```

Executaremos novamente a aplicação no browser e clicaremos sobre o botão "+". Veremos que `linhaDoItem` obteu a `<div>` que contém as informações do `itemId` . De volta ao código JavaScript, nosso próximo passo é modificar a quantidade do item que foi atualizado. Para isso acessaremos `linhaDoItem` , um elemento html, e procuraremos a caixa de texto da quantidade, que por sua vez é um elemento input. Logo,procuraremos na hierarquia html utilizando a função `find()` e utilizaremos como argumento o nome `'input'` . Quando este elemento for encontrado, trocaremos o valor desse elemento utilizando a função jQuery `val` , que receberá o parâmetro o valor da quantidade alterada, isto é, `itemPedido.quantidade`

```
    postQuantidade(data) {
        $.ajax({
            url: '/pedido/updatequantidade',
            type: 'POST',
            contentType: 'application/json',
            data: JSON.stringify(data)
        }).done(function (response) {
                    let itemPedido = response.itemPedido;
                    let linhaDoItem = $('[item-id=' + itemPedido.id +]')
                    linhaDoItem.find('input').val(itemPedido.quantidade);
                    debugger;
                });
    }
}

var carrinho = new Carrinho();
```

Retornaremos ao navegador, na página carrinho. Ao clicarmos nos botões "+" ou "-", a quantidade do item é atualizada automaticamente, no entanto o subvalor total da compra não acompanha essa atualização.

Ao inspecionarmos este elemento, encontraremos um `<span>` que contém um atributo `subtotal` , o que permitirá que encontremos o local em que precisamos fazer a substituição do html.

```
<span class="pull-right" subtotal>
                    149,70
                </span>
```

No código JavaScript, a partir de `linhaDoItem` , localizaremos qual é o elemento que contém um atributo `subtotal` , para isso utilizaremos novamente a função `find()` . Para trocarmos o html de um elemento, utilizamos a função jQuery `.html()` , que receberá como parâmetro o novo valor, ou seja, `itemPedido.subtotal` .

```
    postQuantidade(data) {
        $.ajax({
            url: '/pedido/updatequantidade',
            type: 'POST',
            contentType: 'application/json',
            data: JSON.stringify(data)
        }).done(function (response) {
                    let itemPedido = response.itemPedido;
                    let linhaDoItem = $('[item-id=' + itemPedido.id +]')
                    linhaDoItem.find('input').val(itemPedido.quantidade);
                    linhaDoItem.find('[subtotal]').html(itemPedido.subtotal);
                    debugger;
                });
    }
}

var carrinho = new Carrinho();
```

Ao retornarmos ao browser, veremos que o valor subtotal é alterado à medida que aumentamos ou diminuímos o número de itens no carrinho. A única questão é que o valor está perdendo a formatação correta: ao invés de fazer uso da vírgula ("449,70"), está utilizando o ponto ("449.7") .

Utilizaremos um recurso JavaScript que realiza a modificação de um protótipo númerico. Tal protótipo permitirá que acessemos uma função a partir de um número e assim formatá-lo. Formataremos uma string que possui duas casas decimais, para isso escreveremos `Number.prototype.duasCasas` que será igual a `function()` . No corpo dessa função, declararemos o retorno de uma string formatada, isto é, o valor obtido por meio de `this` . Usaremos o `toFixed()` que retornará duas casas decimais. Em seguida, substituiremos o ponto decimal por uma vírgula utilizando o método `replace()`

```
    postQuantidade(data) {
        $.ajax({
            url: '/pedido/updatequantidade',
            type: 'POST',
            contentType: 'application/json',
            data: JSON.stringify(data)
        }).done(function (response) {
                    let itemPedido = response.itemPedido;
                    let linhaDoItem = $('[item-id=' + itemPedido.id +]')
                    linhaDoItem.find('input').val(itemPedido.quantidade);
                    linhaDoItem.find('[subtotal]').html(itemPedido.subtotal);
                    debugger;
                });
    }
}

var carrinho = new Carrinho();

Number.prototype.duasCasas = function()  {
    return this.toFixed(2).replace('.',',');
}
```

Envolveremos a expressão `(itemPedido.subtotal)` em `()` e utilizaremos a função `duasCasas()`

```
    postQuantidade(data) {
```

```
        postQuantidade(data) {
            $.ajax({
                url: '/pedido/updatequantidade',
                type: 'POST',
                contentType: 'application/json',
                data: JSON.stringify(data)
            }).done(function (response) {
                        let itemPedido = response.itemPedido;

                        let linhaDoItem = $('[item-id=' + itemPedido.id +]')
                        linhaDoItem.find('input').val(itemPedido.quantidade);
                        linhaDoItem.find('[subtotal]').html((itemPedido.subtotal).duasCasas());
                        debugger;
                    });
        }
    }

    var carrinho = new Carrinho();

    Number.prototype.duasCasas = function()  {
        return this.toFixed(2).replace('.',',');
    }
```

Assim feito, a formatação dos itens numéricos da página carrinho estarão corretas.