03

Preparando o Objeto de Resposta Para a Chamada AJAX

Transcrição

Nesta aula iremos resolver mais um problema da nossa aplicação: quando aumentamos a quantidade de itens no carrinho de compras, nós não dispomos da atualização automática da página, o que nos obriga a pressionar o tempo todo a tecla "F5" para vermos os valores corretos. Para deixarmos nossa página dinâmica, modificaremos o JavaScript.

No Visual Studio, abriremos o arquivo `Carrinho.js` dentro da pasta `wwwroot`, subpasta `js`. Nós teremos o método `ajax()` que realiza a chamada para o servidor para que os dados sejam atualizados.

```
class Carrinho {

<****!****>

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

var carrinho = new Carrinho();
```

A página deve ser atualizada no momento em que obtivermos a resposta de sucesso na chamada `ajax()`. Faremo isso ao adicionar uma chamada para outro método, logo depois do `ajax()`. Esse novo método indicará que a chamada foi completada, e seu nome é `done()`, que significa em inglês "completo" ou "terminado". `done()` receberá como parâmetro uma função (`function()`), que por sua vez receberá a resposta do servidor(`response`), enviada por `pedidoController`. Dentro do corpo da função nós inseriremos o código de *refresh* da página, isto é, o comando JavaScript `location.reload()`.

```
class Carrinho {

<****!****>

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

var carrinho = new Carrinho();
```

Feitas as alterações no código, voltaremos a página inicial e selecionaremos um item qualquer para dispormos no carrinho. Já na página de carrinho, atualizaremos o JavaScript pressionando a tecla "F5" E clicaremos sobre o botão "+" como fizemos outras vezes, para avaliar se as modificações surtiram efeito. Notaremos que a quantidade de itens e valores é atualizada automaticamente, contudo esse ainda não é o estado ideal, pois quando fazemos o `location.reload()` estamos obrigando o servidor a recriar toda a página e mandá-la novamente para o navegador, o que aumenta o consumo de banda larga por usuário. O ideal é realizarmos uma chamada `ajax()` que receba uma informação precisa do item que foi alterado e o valor total do carrinho. Deletaremos `location.reload()` e começaremos a montar uma nova classe que fornecerá um objeto de resposta para o método `updateQuantidade()` .

Abriremos o arquivo `PedidoController.cs` , onde teremos o método que fornece a resposta para a chamada `ajax()` , isto é, `UpdateQuantidade()` .

```
 [HttpPost]
        public void UpdateQuantidade([FromBody]ItemPedido itemPedido)
        {
            itemPedidoRepository.UpdateQuantidade(itemPedido);
        }
    }
}
```

`UpdateQuantidade()` não retorna nada, isto é, temos `void` como resposta. O que queremos fazer é retornar um novo objeto. Criaremos uma nova classe chamada `UpdateQuantidadeResponse` dentro da pasta `Models` do nosso projeto. A nova classe irá conter o item de pedido que foi alterado ao clicarmos no botão de "+" ou "-" no carrinho. Esse novo `itemPedido` será apenas de leitura, portanto escreveremos a palavra `get` .

```
namespace CasaDoCodigo.Models
{
    public class UpdateQuantidadeResponse
    {
        public ItemPedido ItemPedido { get; }
    }
}
```

Quando alteramos algum valor, é importante que retornemos todo o carrinho com os valores totais, portanto declararemos uma outra propriedade pública que será o próprio carrinho (`CarrinhoViewModel`), caso seja necessário incluiremos o namespace.

```
namespace CasaDoCodigo.Models
{
    public class UpdateQuantidadeResponse
    {
        public ItemPedido ItemPedido { get; }
        public CarrinhoViewModel CarrinhoViewModel { get; }
    }
}
```

Criaremos um construtor para a classe: selecionaremos as duas propriedades existentes e acionaremos o atalho "Ctrl + ." e a opção "Gerar Construtor".

```
namespace CasaDoCodigo.Models
{
    public class UpdateQuantidadeResponse
    {
        public UpdateQuantidadeResponse(ItemPedido itemPedido, CarrinhoViewModel carrinhoViewModel)
        {
            ItemPedido = itemPedido;
            CarrinhoViewModel = carrinhoViewModel;
        }

        public ItemPedido ItemPedido { get; }
        public CarrinhoViewModel CarrinhoViewModel { get; }
    }
}
```

Dessa forma, temos a classe que fornecerá o objeto de resposta quando o usuário clicar nos botões "+", "-" ou alterar a quantidade de itens diretamente na caixa de texto.