alura

# Extensões do CSS selector do JQuery

Continuando com o projeto, vamos adicionar o botão **Undo** que é usado para trazer os produtos que foram removidos por engano.

Voltando para o nosso código fonte, percebam como está grande a parte do javascript, se isolarmos esse código em algum arquivo, podemos reutilizá-lo em outras páginas quando necessário. Copie todo código e crie um arquivo chamado carrinho.js dentro da pasta do projeto, depois cole todo código javascript neste arquivo. Sabemos como chamar o carrinho.js no nosso html, utilizando a tag `script`:

```
<script src="carrinho.js"></script>
```

Agora no nosso index.html podemos criar o botão **Undo**, logo após a lista de produtos, temos que adicionar um id no botão para acessá-lo de dentro do nosso javascript.

```
<div>
        <input type="button" value="Undo" id="undo">
</div>
```

Com o botão feito podemos voltar no nosso carrinho.js, depois que a página for carregada e alguém clicar no botão uma função será acionada. Portanto temos que adicionar essa função no **aposInicializado** dentro do carrinho.js:

```
$("#undo").click(undo);
```

A função click do Jquery está dizendo que: toda vez que é clicado ele chamará a função undo, que vamos criar agora.

```
var undo = function(){

}
```

Mas como recuperar um produto que removemos completamente da página ? Não é possivel, ao invés de removê-lo, temos que **esconder** o produto da tela, assim podemos mostrar novamente quando o botão é clicado. Então vamos mudar o `removeItem` para usar método `hide`:

```
var removeItem = function(event) {
            event.preventDefault();
```

```
        var self = $(this);
        self.closest("tr").hide();
        atualizaDados();
    };
```

Vejam que toda vez que removemos um produto ele sai da lista mas não é excluido, é possivel identificá-lo na opção de ferramenta para desenvolvedores do Google Chrome, ou pelo firebug do Mozilla Firefox.

Agora na função undo, basta usar o método `show` na `tr` que está escondida. Acessamos através da variável `trs` todas as tags `tr` que estão em hidden, depois usamos o método `show` para mostrar novamente os produtos escondidos:

```
    var undo = function(){
        var trs = $("tr:hidden");
        trs.show();
    }
```

Testamos o botão de **Undo** e ele está funcionando, mas qual produto foi recuperado mesmo?

Vamos fazer com que o produto que foi recuperado fique com uma cor diferente dos outros, para deixar claro que é ele quem voltou. Primeiro criaremos uma classe no carrinho.css que tem o nome de recuperado:

```
.recuperado{
    background:#FE0;
}
```

Agora toda vez que um produto for recuperado temos que adicionar essa classe na tag `tr`, o JQuery tem um método que se chama `addClass(nomeDaClasse)` que faz esse serviço:

```
    var undo = function(){
        var trs = $("tr:hidden");
        trs.addClass("recuperado");
        trs.show();
    }
```

Fazendo alguns testes e funciona!

Percebam que ainda existe um problema. Quando recuperamos, por exemplo, um **Carro**, depois removemos uma **Moto** e logo em seguida recuperamos a **Moto**, os dois produtos (**Carro** e **Moto**) possuem a cor de produto recuperado! Não queremos isso, queremos que fique marcado apenas os produtos que recuperamos quando apertamos **Undo** a última vez (a **Moto** que foi o último produto que recuperamos).

Portanto, sempre que for clicado em Undo **temos** que limpar a classe `recuperar` da lista de produtos e só então adicionar esta classe aos produtos que foram removidos. Mas como fazer isso ? Assim como a função `addClass` o

Jquery possui a função `removeClass` que faz o oposto, então basta usarmos essa função no nosso `undo`:

```
var undo = function(){
    $("tr:visible").removeClass("recuperado");
    var trs = $("tr:hidden");
    trs.addClass("recuperado");
    trs.show();
}
```

Agora ele remove a classe `recuperado` dos produtos da nossa lista que estão visíveis depois adiciona para todos que estão `hidden` e, por fim, mostra eles na tela.

Nossa lista está ok, mas vejam que o valor total e a quantidade de itens não estão atualizando. Porque isso está acontecendo? A nossa função `removeItem` está chamando a função `atualizaDados` que por sua vez começa pegando todos os itens da lista, mesmo que eles estejam hidden. Temos que fazer com que o atualiza dados traga somente os produtos que estão visíveis:

```
var atualizaDados = function(){
        var items = $(".item-total:visible");
            var total = 0;
            for(var i=0; i < items.length; i++) {
                var conteudo = $(items[i]).text();
                var preco = parseFloat(conteudo);
                total += preco;
                }
            $("#valor-total").text(total);
            $("#quantidade-de-itens").text(items.length);

};
```

Vimos que quando usamos um dos seletores do Jquery, podemos passar uma tag (isto é, `<table>`, `<tr>` , `<body>` etc); também podemos usar uma class ou id do css; ou ainda extensões do JQuery para esses seletores, como no caso, o hidden e o visible.

Ver video | Começar exercícios