alura

Selecionando e manipulando elementos

Bem vindo ao nosso curso de manipulação de páginas no cliente através de jQuery. Hoje em dia é bem comum que as páginas sejam mais ricas de informação, mesmo que sem lógica de negócios. É comum vermos páginas onde o resultado de um click é a remoção de um elemento, a criação de uma informação nova ou a alteração de cores, posições etc.

Nesse curso veremos como manipular os elementos de nossa página utilizando o jQuery nas mais diversas situações, entendendo como boas práticas de html, css e javascript andam de mãos dadas.

Começamos o curso baixando nosso projeto através do zip (https://github.com/alura-cursos/jquery-parte2-manipulacao-de-conteudo-dinamico/archive/master.zip) ou forkando (https://github.com/alura-cursos/jquery-parte2-manipulacao-de-conteudo-dinamicoo projeto) o repositório do GitHub.

Abrimos o diretório do nosso projeto e com o duplo clique no arquivo *index.html* visualizamos o mesmo em nosso navegador. Temos um carrinho de compras e todos os produtos serão enviados para São Paulo. Bonito. Se abrirmos o fonte de nosso arquivo vemos que ele usa um arquivo de estilo para deixá-lo com as cores e tamanhos adequados:

```
    <link rel="stylesheet" type="text/css" href="carrinho.css">
```

Perfeito. O que desejamos fazer agora é alterar nossa página para mostrar o valor total de nossa compra. Para isso utilizaremos javascript e o jQuery.

Começamos baixando a versão mais recente do jquery em http://jquery.com/download/ (http://jquery.com/download/). Colocamos o arquivo js no mesmo diretório que nosso projeto. Em nosso html devemos adicionar uma tag de **script** como de costume para incluir código javascript de outro arquivo. Essa tag será inserida como última coisa dentro de nosso *body*:

```
    <script src="jquery-2.0.3.min.js"></script>
</body>
</html>
```

Agora estamos prontos para calcular o total. Primeiro devemos acessar todos os preços de nossa tabela. Olhando o código fonte dela, percebemos que cada linha tem 5 colunas:

```
                <tr>
                    <td><img src="imagens/cinto.jpg"></td>
                    <td>Cinto 42</td>
                    <td>1</td>
```

```
            <td>42.00</td>
            <td>42.00</td>
        </tr>
```

Estamos interessados na quinta coluna. A coluna que representa o valor **total** daquele **item** (seu preço unitário vezes a quantidade). Mas como poderemos acessá-lo? Vamos dar um nome para essa tag, mas não pode ser um nome único pois a mesma tag aparece para cada linha, para cada produto. O nome que daremos é **item-total**. Como esse nome é uma característica de diversas tags, usamos uma classe css para chegar nesse resultado:

```
<td class="item-total">42.00</td>
```

Alteramos todas as quintas colunas para colocar essa classe. Agora sabemos que quando estivermos interessados nos valores totais dos itens, basta acessarmos as tags que tem essa classe.

Vamos então ao Javascript com jQuery. O selecionador de classes é o caracter . (ponto), portanto podemos selecionar todos os **item-total** fazendo:

```
.item-total
```

Mas queremos usar isso através do jquery:

```
$(".item-total");
```

E ele nos retorna todas essas tags:

```
<script>
var items = $(".item-total");
</script>
```

Queremos passar por todos os elementos dessa array e para isso basta fazer um laço:

```
for(var i=0; i < items.length; i++) {
}
```

Precisamos agora extrair o valor de cada um desses itens com o método **text**:

```
for(var i=0; i < items.length; i++) {
    var conteudo = $(items[i]).text();
}
```

Como o método **text** devolve o valor como *String*, faremos a conversão para *float*, ponto flutuante:

```
for(var i=0; i < items.length; i++) {
    var conteudo = $(items[i]).text();
    var preco = parseFloat(conteudo);
}
```

Pronto, basta somar todos os valores:

```
var total = 0;
for(var i=0; i < items.length; i++) {
    var conteudo = $(items[i]).text();
    var preco = parseFloat(conteudo);
    total += preco;
}
```

Agora vamos criar uma *div* que terá o valor total em nossa tela. Dentro do nosso html, logo antes de definir a tag **script** (após finalizar a última tag **div** da página), adicionamos uma nova *div* com o valor total:

```
<div>
    Valor total:
</div>
```

Mas qual o valor total antes de calcularmos? Zero.

```
<div>
    Valor total: R$0
</div>
```

Ainda tem algo de errado aqui. Quando alterarmos o valor total, desejamos alterar somente aquele número 0, não tudo. Então vamos adicionar uma tag para encapsular somente o preço:

```
<div>
    Valor total: R$<span id="valor-total">0</span>
</div>
```

Agora sim temos que o valor do nosso carrinho está em uma tag que conseguimos acessar. Ele está isolado e ele tem um nome, um nome único, seu id: **valor-total**.

```
$("#valor-total").text(total);
```

Perfeito. Mas e a quantidade de itens distintos? Começamos agora adicionando um novo elemento à nossa div:

```
<br/>
Quantidade de itens distintos:
<span id="quantidade-de-itens"></span>
```

Para extrair o tamanho da array, utilizamos o atributo *length*, e podemos escrevê-lo diretamente em nosso *span*:

```
$("#quantidade-de-itens").text(items.length);
```

Nosso código final fica:

```
<script>
    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);
</script>
```

Salvamos o arquivo, atualizamos nosso navegador e temos o resultado que queríamos: tanto o preço quanto a quantidade de itens.

Mas o que acontece se o nosso código *script* vier antes de nossa última *div*? Movemos ele de lugar e atualizamos a página.

Agora fica tudo em branco! Como assim? O que acontece é que o código javascript foi executado antes das tags existirem. Como as tags não existem, o jquery não retorna nenhum elemento, e não aplica o resultado da soma e contagem em nenhum elemento. Repare que para evitar esse tipo de situação queremos ter certeza que o javascript seja executado após todas as tags terem sido carregadas. Como fazer isso?

O jQuery permite fazer isso. Basta criarmos uma função de inciialização:

```
var aposInicializado = function() {
    var items = $(".item-total");
    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);
    };
```

E dizer para o jQuery invocá-la:

```
    $(aposInicializado);
```

Salvamos, atualizamos a página e agora tudo funciona novamente! Nosso código javascript somente é executado após a página ser carregada. Por educação, colocaremos o javascript após nossa div, mas já sabendo de antemão que a boa prática é manter a função de inicialização e o código javascript no final.

Em breve veremos como remover itens do carrinho e voltar atrás em nossas modificações.

Ver video | Começar exercícios