alura

Criando elementos dinamicamente

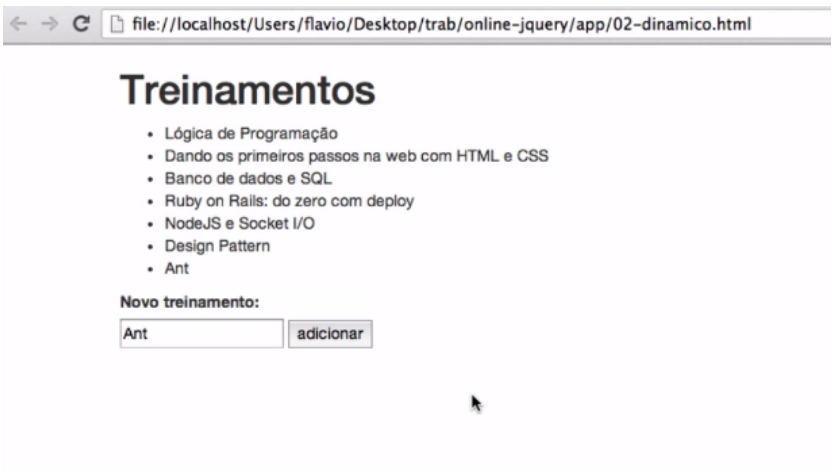
Revisão

Vimos até agora como manipular elementos com jQuery, inclusive como executar modificações em nossa página a partir das ações do usuário, como por exemplo, o click de um botão. Tudo que fizemos até agora foi manipular elementos já existentes.

Mas o que fazer quando precisamos adicionar novos elementos? É justamente isso que iremos aprender neste vídeo.

jQuery como fábrica de elementos

Vejamos uma página que exibe uma lista de treinamentos, onde cada item, ao ser clicado, exibe um alerta com o texto com seu nome. O usuário pode adicionar quantos treinamentos quiser na lista. Vamos implementar essas funcionalidades.



Temos um HTML com uma nova estrutura.

```
<html lang="pt">
<head>
    <meta charset="UTF-8">
    <title>Introdução ao jQuery</title>
    <link rel="stylesheet" href="css/bootstrap.css">
    <link rel="stylesheet" href="css/estilos.css">
</head>
<body class="container">
    <h1>Treinamentos</h1>
    <ul id="lista">
        <li>Lógica de Programação</li>
```

```
            <li>Dando os primeiros passos na web com HTML e CSS</li>
            <li>Banco de dados e SQL</li>
            <li>Ruby on Rails: do zero com deploy</li>
            <li>NodeJS e Socket I/O</li>
        </ul>
        <label>Novo treinamento:</label>
        <input id="treinamento">
        <input id="botao-adiciona" type="button" value="adicionar"/>
        <script src="js/jquery.js"></script>
        <script>
            //aqui vamos mexer
        </script>
    </body>
    </html>
```

Pediremos ao jQuery através de um seletor de TAG do CSS que selecione o elemento que queremos manipular, neste caso, este seletor nos retornará todos os itens da lista. Agora, através da função click, precisamos obter o texto do item clicado.

```
<script>
    $('li').click(function() { //evento click para cada elemento li

    });
</script>
```

Como saberemos neste trecho de código qual item foi clicado? O primeiro, o segundo, o último? Assim como no JavaScript puro, usamos a palavra chave "this" para termos acesso ao elemento que disparou o evento. A cada click, "this" adota um valor diferente. Se clicarmos na segundo item da lista, "this" será este item. Nos resta apenas chamar a função text() para retornar o texto do item, mas isso não funcionará. Isto porque, this não é um elemento do jQuery, mas um elemento nativo do JavaScript que não possui todas as facilidades do jQuery. Podemos promove-lo, isto é, transformá-lo num objeto do jQuery envolvendo-o com a função dólar. Agora é só chamar a função text(). A clássica função alert se encarregará de exibir o texto do item na tela.

```
<script>
    $('li').click(function() {
        var texto = $(this).text();
        alert(texto);
    });
</script>
```

Ao testar o resultado, tudo funciona conforme esperado.

Queremos adicionar dinamicamente elementos na lista e o código que fará isso será disparado pelo botão da página. Então, primeiro adicionaremos o evento click nele.

```
$('#botao-adiciona').click(function() {

});
```

Agora, precisamos capturar o texto digitado, o que não é novidade para nós. Com jQuery , selecionamos o input com ID 'treinamento' para logo em seguida guardar na variável texto o seu valor através da função val().

```
$('#botao-adiciona').click(function() {
    var texto = $('#treinamento').val();
});
```

Pronto, já temos o que foi digitado pelo usuário. Falta criar o item da lista para associarmos em seguida o que foi digitado. A função dólar, além de selecionar o elemento do documento, também é uma fábrica de elementos. A diferença é que, quando queremos criar elementos, no lugar de passarmos um seletor CSS, passamos a tag correspondente que queremos criar.

```
$('#botao-adiciona').click(function() {
    var texto = $('#treinamento').val();
        $('<li>');
});
```

Pronto, temos um novo item da lista, e, como todo objeto do jQuery, podemos adicionar o texto capturado através da função Text. Temos o elemento criado e agora com conteúdo, mas ele ainda não faz parte do documento. Incluiremos o elemento como último item da lista por meio da função appendTo, passando como parâmetro o seletor CSS correspondente a lista de nossa página, a

- com id "lista".

```
$('#botao-adiciona').click(function() {
    var texto = $('#treinamento').val();
        $('<li>').text(texto).appendTo('#lista');
});
```

Pronto, já podemos testar.

Agora, podemos adicionar quantos treinamentos quisermos, que a lista da página será atualizada dinamicamente.

Tempos um problema. Os novos item, ao serem clicados, não exibem o alerta, porque foram adicionados após a atribuição do evento click.

Mecanismo de delegação de Eventos

Podemos resolver este problema usando o mecanismo de delegação do jQuery através da já conhecida função on(). Toda vez que uma li for clicada, mesmo sendo recém adicionada, quem responderá ao evento será a lista, mas levando em consideração o elemento que foi clicado. Para usar essa mecanismo, a função on() muda um pouco. Como seletor, selecionamos a lista, por fim, adicionamos mais um parâmetro na função on() que indica a origem do evento, em nosso caso, qualquer li.

```
<script>
    $('#lista').on('click', 'li', function() {
        var texto = $(this).text();
        alert(texto);
    });
</script>
```

Pronto, testando mais uma vez, vemos que os itens adicionados dinamicamente estão exibindo o alerta.

Ver video | Começar exercícios