Este curso é de propriedade do aplicativo SoloLearn e foi traduzido por Ana Beatriz Augusto usando os recursos Reverso Context e Google Tradutor.

Lidando com eventos

O jQuery fornece uma maneira eficiente de lidar com eventos. Eventos ocorrem quando o usuário faz uma ação, como clicar em um elemento, mover o mouse, ou enviar um formulário.

Quando um evento ocorre em um elemento, uma função é executada.

Por exemplo, vamos supor que queremos lidar com o evento `click` num elemento que tem `id="demo"` e mostrar a data atual quando o botão é clicado.

Usando JavaScript puro, o código ficaria assim:

```
var x = document.getElementById("demo");
x.onclick = function(){
  document.body.innerHTML = Date();
}
```

Poderíamos lidar com o mesmo evento com este código jQuery:

```
$("#demo").click(function(){
  $("body").html(Date());
});
```

Como podemos ver, o código jQuery é mais curto e mais fácil de se ler e escrever.
Note que, não é usado **on** no nome do evento (`onclick` no JavaScript é `click` no jQuery)

Eventos Comuns

Estes são os eventos mais usados:

Eventos de mouse:

`click` ocorre quando um elemento é clicado.

`dblclick` ocorre quando se clica duas vezes no elemento.

`mouseleave` ocorre quando o ponteiro do mouse sai de cima do elemento.

Eventos do teclado:

`keydown` ocorre quando uma tecla é pressionada.

`keyup` ocorre quando uma tecla é solta.

Eventos de formulário:

`submit` ocorre quando um formulário é enviado.

`change` ocorre quando o valor de um elemento foi mudado.

`focus` ocorre quando um elemento é focado.

`blur` ocorre quando um elemento perde o foco.

Eventos de documento:

`ready` ocorre quando o DOM foi carregado.

`resize` ocorre quando a janela do navegador muda de tamanho.

`scroll` ocorre quando o usuário rola em um elemento específico.

Como exemplo, vamos mudar o conteúdo de uma `div` quando o usuário digita em um `input`. Para isso, precisamos lidar com o evento `keydown`, o qual ocorre quando uma tecla é pressionada:

HTML:

```
<input type="text" id="name" />
<div id="msg"></div>
```

JavaScript:

```
$("#name").keydown(function(){
  $("#msg").html($("#name").val());
});
```

O código lida com o evento `keydown` no elemento com `id="name"` e coloca o que foi escrito no `input` como conteúdo da `div` com `id="msg"`.

Lidando com eventos

Outra maneira de se lidar com eventos é usando o método `on()`.
O método `on()` é usado para atribuir um evento ao elemento selecionado. Por exemplo:

```
$("p").on("click", function(){
  alert("clicado");
});
```

Como podemos ver, o nome do evento é passado como primeiro parâmetro no método `on()`. O segundo parâmetro é a função.

O método `on()` é útil para unir a mesma função a múltiplos eventos. Você pode escrever vários nomes de eventos separados por espaços como primeiro parâmetro. Por exemplo, você pode usar a mesma função para os eventos `click` e `dblclick`.

O método off()

Você pode remover eventos usando o método off().

Por exemplo:

```
$("div").on("click", function(){
  alert("Olá!");
  $("div").off("click");
});
```

O parâmetro do método off() é o nome do evento que deseja remover.

Lista de afazeres

Vamos criar uma lista de afazeres usando os conceitos mostrados até aqui.
A lista de afazeres terá a funcionalidade de adicionar novos itens na lista e de apagar itens já existentes.
Primeiro, vamos criar o HTML:

```
<h1> Minha lista de afazeres </h1>
<input type="text" placeholder="Novo item" />
<button id="add">Adicionar</button>
<ol id="mylist"></ol>
```

Clicar no botão adiciona o que foi escrito no `input` na lista.

Agora que nosso HTML está pronto, podemos começar a escrever nosso código jQuery.
Primeiro, colocamos uma função para lidar com o evento `click` no botão:

```
$(function(){
  $("#add").on("click",
function(){
    // Código
  });
});
```

Dentro da função, vamos selecionar o que foi digitado (o valor do elemento `input`) e criar um novo elemento `li`, adicionando ele na lista:

```
var val = $("input").val();
if(val !== '') {
  var elem =
$("<li></li>").text(val);
  $(elem).append("<button
class='rem'> X </button>");
  $("#mylist").append(elem);
  $("input").val(""); //
Limpando o input
}
```

O código da página anterior pega o que foi digitado e coloca na variável `val`. A condição `if` checa se o que foi digitado está em branco e então cria um novo elemento `li`. Um botão de remoção é adicionado depois do novo elemento criado ser adicionado a lista `<ol id="mylist">`.
Aqui está o código completo:

```
$(function(){
  $("#add").on("click", function(){
    var val = $("input").val();
    if(val !== '') {
      var elem = $("<li></li>").text(val);
      $(elem).append("<button class='rem'>X</button>");
      $("#mylist").append(elem);
      $("input").val("");
    }
  });
});
```

O que resta fazer é lidar com o evento `click` no botão com `class="rem"` e remover o correspondente elemento `li` da lista.

```
$(".rem").on("click", function(){
  $(this).parent().remove();
});
```

Lembre-se que `this` é o objeto atual. O código acima remove o pai do objeto atual, que no nosso caso é o elemento `li`.

O código completo:

```
$(function(){
  $("#add").on("click", function(){
    var val = $("input").val();
    if(val !== '') {
     var elem = $("<li></li>").text(val);
     $(elem).append("<button class='rem'>X</button>");
     $("#mylist").append(elem);
     $("input").val("");
     $(".rem").on("click", function() {
      $(this).parent().remove();
     });
    }
  });
});
```

Este curso é de propriedade do aplicativo SoloLearn e foi traduzido por Ana Beatriz Augusto usando os recursos Reverso Context e Google Tradutor.