

Criando um botão remover

Transcrição

Agora que estamos conseguindo adicionar item a nossa tabela placar, também precisamos dar a opção do usuário de remover alguma das linhas que ele desejar.

Para isto, vamos adicionar um pequeno botão de remover em cada linha da tabela, como uma terceira coluna. Ele será um link representado por um ícone de lixeira, que pegaremos do framework que estamos utilizando para estilizar, o *Materialize*. Vamos adicioná-lo dentro de um `<td>`, como uma nova coluna da tabela:

```
<section class="placar">
  <h3>Placar</h3>
  <table>
    <thead>
      <tr>
        <th>Usuário</th>
        <th>No. de palavras</th>
        <th>Remover</th>
      </tr>
    </thead>
    <tbody>
      <tr>
        <td>Padrao</td>
        <td>999</td>
        <td>
          <a href="#">
            <i class="small material-icons">delete</i>
          </a>
        </td>
      </tr>
      <tr>
        <td>Usuario Exemplo</td>
      </tr>
    </tbody>
  </table>
</section>
```

Mas deixando-o estático, ele irá aparecer apenas nas pontuações que já vem no `principal.html`. Vamos alterar a nossa função `inserePlacar()` para adicionar um botão de remoção na linha também:

```
function inserePlacar() {
  var corpoTabela = $(".placar").find("tbody");
  var usuario = "Douglas"
  var numPalavras = $("#contador-palavras").text();
  var botaoRemover = "<a href='#><i class='small material-icons'>delete</i></a>" ;

  var linha = "<tr>" +
    "<td>" + usuario + "</td>" +
    "<td>" + numPalavras + "</td>" +
    "<td>" + botaoRemover + "</td>" +
    "</tr>";
```

```
        corpoTabela.append(linha);  
    }  
}
```

Primeiro, criaremos o botão com uma string, depois, o adicionaremos como uma terceira coluna da linha.

Com o botão aparecendo na nossa página tanto nas linhas que já vem no placar quanto as que são criadas no fim do jogo, vamos começar a adicionar um evento de click em nossos botões:

Primeiramente, vamos adicionar uma classe à tag `<a>` :

```
<a href="#" class="botao-remover">  
    <i class="small material-icons">delete</i>  
</a>
```

E vamos adicionar um evento de click no `main.js` :

```
$(".botao-remover").click(function(){  
    console.log("Fui clicado");  
})
```

Prevenindo o salto da tela

Repare que ao clicar no ícone de remoção, além de exibir a mensagem no console a página salta para o topo. Isso acontece pois quando clicamos em uma tag `<a>` ela tem o comportamento de redirecionar para o que estiver em seu atributo `href`, seja um link ou um ID de um elemento HTML. Neste caso, estamos com o valor de `"#"` no `href`, o que faz com que seja redirecionado pro topo da página. Se o `href` fosse o id de algum elemento, a visualização iria saltar para este elemento.

Este é um comportamento padrão da tag `<a>`, mas neste caso, queremos evitá-lo. Conseguimos fazer isto através da função `preventDefault()`. Vamos alterar nosso evento para receber um parâmetro que permite usá-la:

```
$(".botao-remover").click(function(event){  
    event.preventDefault();  
    console.log("Fui clicado");  
})
```

Experimente clicar no botão agora, você verá que apenas a mensagem é exibida e a tela parou de saltar.

Removendo a linha

Agora que estamos detectando o evento de click, partiremos para a remoção da linha. Para remover um elemento do HTML, podemos utilizar a função com nome semelhante do jQuery, a `.remove()`.

Se no nosso evento de click, utilizarmos ela para remover quem foi clicado, que pode ser acessado através do **this**:

```
$(".botao-remover").click(event, function(){  
    this.remove();  
})
```

```
})
```

Vamos obter um **erro clássico** de quando estamos trabalhando com jQuery. Tentamos utilizar uma função do **jQuery** em um **elemento do HTML**. Elementos comuns do HTML não tem acesso as funções do jQuery, precisamos empoderá-los com as funções do objeto jQuery, envolvendo-os com a função jQuery:

```
$(".botao-remover").click(event, function(){  
    $(this).remove();  
})
```

Deste modo o `this` (elemento HTML que foi clicado), ganha acesso as várias funções do jQuery, como `.text()` , `.val()` , `.css()` e todas as outras funções do jQuery.

Só que o código acima não resolve nosso problema, se você clicou no botão remover, você vai ver que apenas o ícone de lixeira sumiu! A linha com a pontuação do usuário continua lá. Isto acontece pois estamos chamando a função **.remove()** na tag `<a>` , que é quem foi clicada!

O que nós desejamos é clicar na tag `<a>` e **remover** a linha inteira, que é a `<tr>` ! Só que a `<tr>` é um elemento **acima** da tag no HTML , veja:

```
<tr>  
  <td>Padrao</td>  
  <td>999</td>  
  <td>  
    <a href="#">  
      <i class="small material-icons">delete</i>  
    </a>  
  </td>  
</tr>
```

O elemento **pai** da `<a>` é uma `<td>` , e a **linha** é o elemento dois níveis acima, ou seja , a `<tr>` . Para acessarmos um elemento **acima** do elemento selecionado com jQuery, um elemento pai, temos a função `.parent()` do Javascript, veja:

```
$(".botao-remover").click(){  
    $(this).parent().remove();  
}
```

O elemento `$(this).parent()` é o `<td>` , como queremos remover a linha inteira, que é o `<tr>` , vamos subir mais um nível na árvore do HTML utilizando a função `.parent()` duas vezes:

```
$(".botao-remover").click(){  
    $(this).parent().parent().remove();  
}
```

Agora sim nossa linha é removida corretamente quando clicamos no tag `<a>` !

O que aprendemos:

- A prevenir um comportamento comum com o `event.preventDefault()`.
- Como envolver um elemento do HTML com as funções do jQuery
- A função `.remove()` para remover um elemento do HTML
- A função `.parent()` do jQuery, para subir na árvore do HTML