07

Query Selector

Transcrição

Agora que temos o domínio das funções iniciais do JavaScript, `alert()` e `console.log()`, usadas para exibir mensagens ao usuário, vamos começar a manipular os dados do site da Aparecida. A página deixará de ser estática e ganhará mais interatividade. O índice de massa corporal (IMC) será calculado automaticamente, e conseguiremos adicionar um novo paciente sem alterarmos o HTML. Isso dará dinamismo e deixará a página mais moderna.

Para começarmos a manipular a página, se quiséssemos alterar o título "Meus pacientes" para "Meus clientes", como isso poderia ser feito com JavaScript? Precisaríamos ter acesso ao código do arquivo HTML. Tudo o que estiver contido na tag `<script>` será interpretado como JS e, o que está fora, como HTML. Teremos que levar as funcionalidades criadas em HTML para o mundo JS.

Primeiramente, vamos conhecer o **DOM** (Document Object Model), representação do HTML para o nosso JavaScript, acessível por uma palava do JavaScript chamada `document`.

No console do navegador, quando digitamos `document` e pressionamos "ENTER" em seguida, veremos **tudo** o que está na página HTML. O `document` será a ponte entre o JavaScript e o HTML, e tudo que for alterado nele será alterado na exibição para o usuário.

Vamos experimentar adicionar o `document` na tag `<script>`, lembrando que não iremos usá-lo entre `""` (aspas), pois ele funcionará como uma varíavel, e as aspas são usadas quando trabalhamos com uma *string*, e não é uma palavra ou frase que queremos imprimir.

```
<head>
    <meta charset="UTF-8">
    <title>Aparecida Nutrição</title>
    <link rel="icon" href="favicon.ico" type="image/x-icon">
    <link rel="stylesheet" type="text/css" href="css/reset.css">
    <link rel="stylesheet" type="text/css" href="css/index.css">

    <script>
        console.log("Oi Mundo");
        console.log(document);
    </script>

</head>
```

Se voltarmos ao navegador e atualizarmos a página, veremos que no JavaScript teremos acesso ao `document`. Conseguiremos ver o seguinte código no navegador:

```
#document
    <!DOCTYPE html>
    <html lang="pt-br">
        <head>...</head>
        <body>...</body>
    </html>
```

Mas se não quisermos manipular o DOM inteiro, e sim apenas um pedaço, por exemplo, o texto dentro da tag `<h1>`, localizada acima do fechamento do `<header>`,

```
<header>
    <div class="container">
        <h1>Aparecida Nutrição</h1>
    </div>
</header>
<main>
    <section class="container">
        <h2>Meus paciente</h2>
        <table>
            <thead>
//...
```

como faríamos para modificar apenas o texto "Aparecida Nutrição", que é um pedaço do `document`?

Vamos encontrar uma forma de **pesquisar** somente a tag `<h1>`. Para isto, usaremos o método `querySelector()`, passando como parâmetro o que queremos encontrar - neste caso, entre aspas, pois queremos o termo exato. No console, iremos digitar:

```
document.querySelector("h1");
```

Após executarmos o método, ele retornará o conteúdo da tag:

```
document.querySelector("h1");
    <h1>Aparecida Nutrição</h1>
```

Assim, será selecionado o primeiro `h1` da página, justamente aquele que queremos modificar. Então podemos passar este código para o navegador e imprimir o `h1` no console do navegador para verificarmos se ele realmente foi selecionado, utilizando o `console.log()` novamente dentro da tag `<script>`:

```
<script>
    console.log(document.querySelector("h1"));
</script>
```

Mas ao atualizarmos a página, o retorno será `null`. Por que isso acontece? Se usamos o `querySelector()` no console, conseguimos que o `h1` seja retornado, porém, isto não ocorre no código. Qual é a diferença?

O browser, ao carregar a página HTML, vai lendo linha por linha, de cima para baixo. Quando ele chega na tag `<script>`, ele tenta buscar um `h1` na página, porém, isto não está carregado em sua memória. A tag `<h1>` está **abaixo** do código JavaScript e ainda não foi interpretado pelo navegador, logo, ele não poderá ser selecionado.

Até agora estamos escrevendo HTML e JavaScript no mesmo arquivo, o que pode se tornar confuso conforme nosso código for crescendo. Para evitarmos isso, poderemos escrevê-los em arquivos separados.

O mesmo não ocorre quando executamos o código no console do navegador, pois a página já estará totalmente carregada, e o `document` estará completo.

Por conta de situações como esta, é uma boa prática colocar a tag `<script>` no fim do HTML, mais precisamente, como último elemento de `<body>` após o fechamento de `<main>`:

```
<!-- ... -->
        </section>
    </main>
<script>
    console.log(document.querySelector("h1"));
</script>
</body>
```

Desta vez, ao recarregarmos a página, o `h1` é impresso no console do navegador, sendo selecionado corretamente.

Agora que conseguimos selecionar o `h1`, o que é preciso fazer para alterarmos o texto? Primeiramente, em vez de imprimir, salvaremos a parte selecionada, no caso o `h1`, dentro da variável `titulo`. Para isso, usaremos a palavra `var`:

```
<!-- ... -->
        </section>
    </main>
<script>
    var titulo = document.querySelector("h1");
    console.log(titulo);
</script>
</body>
```

No console, a tag `<h1>` continuará sendo exibida.

```
<h1>Aparecida Nutrição</h1>
```

Porém, nosso real objetivo é pegar o conteúdo textual `Aparecida Nutrição`.

Algumas tags, como `h1`, `h2`, `p` e `span`, possuem um **conteúdo de texto**. Nesses casos, o JavaScript possui uma propriedade que nos permite acessá-lo: `textContent`. Vamos testar e imprimir o conteúdo de texto da variável `titulo`, que representa o `h1`:

```
<!-- ... -->
<script>
    var titulo = document.querySelector("h1");
    console.log(titulo);
    console.log(titulo.textContent);
</script>
```

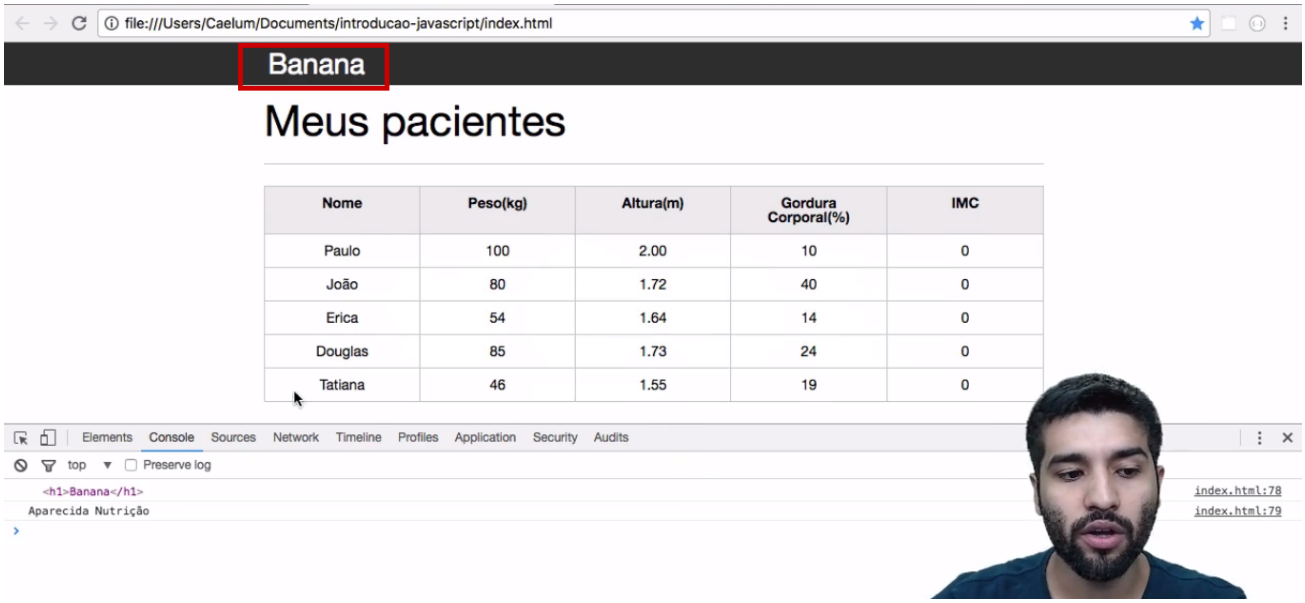
De volta ao navegador, veremos uma diferença no que será impresso pelos dois `console.log()`s:

```
   <h1>Aparecida Nutrição</h1>
Aparecida Nutrição
```

Somente o texto **Aparecida Nutrição** será impresso no segundo `console.log()`. Então conseguiremos acessar e exibir o conteúdo de texto da tag. E para alterá-lo, basta usar o `textContent` e passar um novo valor para o `titulo`, igualando-o a um novo texto:

```
<script>
    var titulo = document.querySelector("h1");
    console.log(titulo);
    console.log(titulo.textContent);

    titulo.textContent = "Banana";
</script>
```



Observe que o título da página foi trocado. Se quisermos alterá-lo novamente, por exemplo, para "Aparecida Nutricionista", basta modificar a propriedade `textContent`.

```
<script>
    var titulo = document.querySelector("h1");

    titulo.textContent = "Aparecida Nutricionista";
</script>
```

Assim, nós alteramos o conteúdo de texto.