01

Replicando a validação e o cálculo do IMC para todos os pacientes

Transcrição

Conseguimos realizar a validação do primeiro paciente, calculando o IMC e verificando se a altura e o peso são válidos. A seguir, replicaremos esse código de validação e cálculo do IMC para os demais pacientes da tabela, como no trecho abaixo:

```
<tbody id="tabela-pacientes">
    <tr class="paciente" id="primeiro-paciente">
        <td class="info-nome">Paulo</td>
        <td class="info-peso">1000</td>
        <td class="info-altura">2.00</td>
        <td class="info-gordura">10</td>
        <td class="info-imc">0</td>
     </tr>
//...
```

O código responsável por calcular o IMC do paciente e as validações dos dados é, praticamente, todo o arquivo `principal.js`. Então, bastaria copiarmos esse código, modificar a classe `primeiro-paciente` e os nomes das variáveis, que já validaríamos outro paciente.

```
var paciente2 = document.querySelector("#segundo-paciente");

var tdPeso2 = paciente.querySelector(".info-peso");
var peso2 = tdPeso.textContent;

var tdAltura2 = paciente.querySelector(".info-peso");
var altura2 = tdAltura.textContent;
```

No entanto, a **repetição de código** não é uma boa prática de programação, o ideal é sempre reaproveitar o trabalho já pronto. No caso, poderíamos fazer um *loop* - conhecido na lógica de programação -, para conseguirmos iterar por cada linha da tabela, ou seja, por cada paciente. Desta forma, selecionaremos os dados de cada um deles e calcularemos os IMCs.

Então, se quisermos iterar por todos os pacientes, devemos selecioná-los - e não somente um deles, como estamos fazendo atualmente. Se analisarmos o código da tabela no `index.html`, veremos que **todos os pacientes** possuem a classe `paciente`:

```
<tbody id="tabela-pacientes">
    <tr class="paciente" id="primeiro-paciente">
        <td class="info-nome">Paulo</td>
        <td class="info-peso">-100</td>
        <td class="info-altura">4.00</td>
        <td class="info-gordura">10</td>
        <td class="infor-imc">0</td>
    </tr>
    <tbody id="tabela-pacientes">
```

```
    <tr class="paciente" id="paciente">
        <td class="info-nome">João</td>
        <td class="info-peso">80</td>
        <td class="info-altura">1.72</td>
        <td class="info-gordura">40</td>
        <td class="info-imc">0</td>
    </tr>
<!-- ... -->
```

Logo, em vez de buscarmos pelo id `primeiro-paciente`, vamos buscar pela classe `paciente`. Testaremos isso:

```
var pacientes = document.querySelector(".paciente");
console.log(pacientes);
```

Apesar de termos selecionado por classe, só nos foi retornado o primeiro paciente, a primeira linha com a classe `paciente`. Isso porque a função `querySelector()` nos retorna apenas **um elemento**, independentemente do que passarmos para a mesma. Se estivermos interessados em buscar vários elementos com a classe `.paciente`, devemos utilizar `querySelectorAll()`:

```
var pacientes = document.querySelectorAll(".paciente");
console.log(pacientes);
```

Essa função nos retorna um array no console, com todos os elementos que possuem a classe `paciente`.

```
[tr#primeiro-paciente.paciente, tr.paciente, tr.paciente, tr.paciente, tr.paciente]
```

Ao abrirmos o array, veremos todos os pacientes:

```
NodeList[5]
    0: tr#primeiro-paciente.paciente
    1: tr.paciente
    2: tr.paciente
    3: tr.paciente
    4: tr.paciente
    length: 5
```

Veremos impresso também a propriedade `length` que nos informa que temos `5` elementos.

Iterando sobre os pacientes

Agora que temos todos os pacientes da tabela, devemos iterar por eles, e para cada um deles executaremos o código responsável por calcular o IMC e validar o peso e a altura.

Há várias formas de fazermos isso, e uma delas, bastante conhecida por todo mundo que já viu algo de lógica de programação é o *loop* `for`. Ele receberá três argumentos: a declaração da variável inicial, até onde queremos que essa variável cresça, e o que queremos fazer no fim de cada iteração. Por exemplo:

```
var pacientes = document.querySelectorAll(".paciente");

for (var i = 0; i < 5; i++) {

}
```

Mas não queremos percorrer um número fixo, e sim percorrer a lista de pacientes. Todo *array* possui uma propriedade chamada `length`, que nos informa o seu tamanho. Então vamos usá-la para percorrer até o tamanho da lista de pacientes:

```
var pacientes = document.querySelectorAll(".paciente");

for (var i = 0; i < pacientes.length; i++) {
    console.log()

}
```

Deste modo, evitaremos trabalhar com um tamanho fixo de elementos.

Cada elemento do array tem um índice, que começa da posição `0`, o qual representará o primeiro paciente, o `1` será o segundo, e assim por diante. Para acessarmos o paciente do array, basta passarmos o índice entre colchetes para o mesmo. Esse índice será a variável `i`, que percorrerá do valor `0` até a última posição do array de pacientes **menos** `1`:

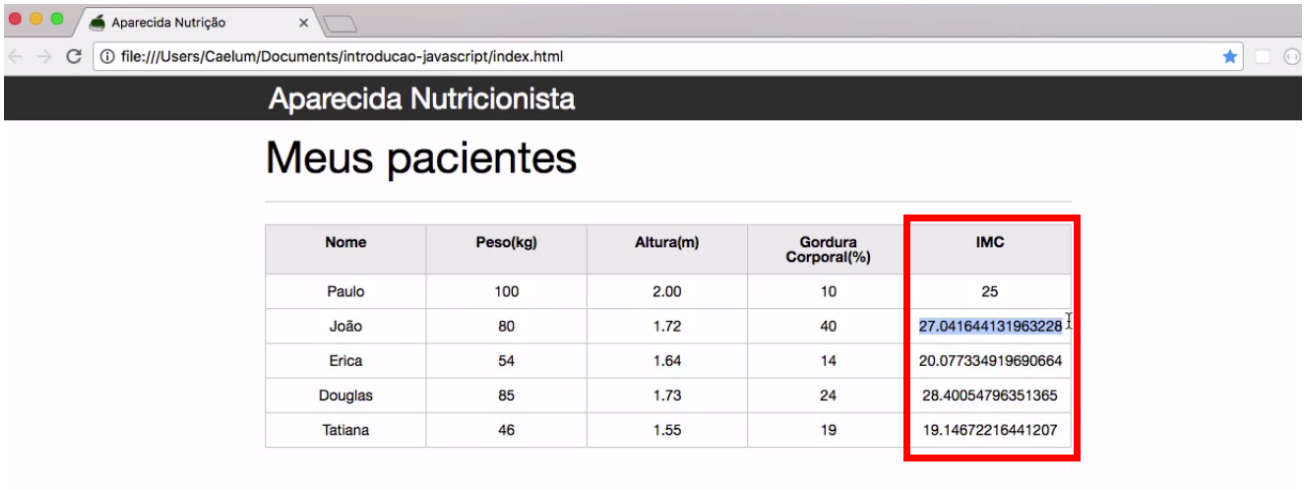
```
var pacientes = document.querySelectorAll(".paciente");

for (var i = 0; i < pacientes.length; i++) {
    console.log(pacientes[i]);
}
```

Feito isso, temos acesso ao `pacientes[i]`, e bastará movermos o código que calcula o IMC e faz a validação para dentro do `for`. Além disso, faremos um ajuste para não precisarmos alterar todo o código. Usaremos um pequeno "truque", criando a variável `paciente`, que será um atalho para `pacientes[i]`:

```
var pacientes = document.querySelectorAll(".paciente");

for (var i = 0; i < pacientes.length; i++) {

    var paciente = pacientes[i];

    var tdPeso = paciente.querySelector(".info-peso");
    var peso = tdPeso.textContent;

    var tdAltura = paciente.querySelector(".info-altura");
    var altura = tdAltura.textContent;

    var tdImc = paciente.querySelector(".info-imc");

    var pesoEhValido = true;
    var alturaEhValida = true;

    if (peso <= 0 || peso >= 1000) {
```

```
            console.log("Peso inválido!");
            pesoEhValido = false;
            tdImc.textContent = "Peso inválido";
        }

        if (altura <= 0 || altura >= 3.00) {
            console.log("Altura inválida!");
            alturaEhValida = false;
            tdImc.textContent = "Altura inválida";
        }

    if (alturaEhValida && pesoEhValido) {
            var imc = peso / (altura * altura);
            tdImc.textContent = imc;
        }
}
```

De volta ao browser, veremos que conseguimos calcular o IMC de todos os pacientes!



Limitando as casas decimais de um número

Alguns dos IMCs possuem diversas casas decimais, tornando o número imenso. Podemos melhorar isso utilizando uma função que limita esta quantidade. Essa função é `toFixed()`, que recebe como parâmetro a quantidade de casas decimais a serem exibidas depois do ponto.

Vamos utilizá-la no momento em que o IMC é impresso na célula:

```
if (alturaEhValida && pesoEhValido) {
    var imc = peso / (altura * altura);
    tdImc.textContent = imc.toFixed(2);
}
```

Agora a visualização dos IMCs ficou um pouco melhor, sem tantos números nas casas decimais.

Vimos um *loop* simples, bastante utilizado na lógica de programação. Com o `for`, conseguimos validar o IMC e fazer os cálculos para todos os pacientes. Se os dados de algum paciente estiverem incorretos, um aviso será exibido na célula relacionada da tabela.