04

Criando um objeto paciente

Transcrição

Continuando com as melhorias e refatorações do nosso código e aplicando-se boas práticas, seria interessante quebrarmos o código do `form.js` em funções menores, considerando que atualmente temos uma função com várias responsabilidades, como capturar os valores do formulário, criar a `tr` e as `td` s do paciente, colocar os valores nas linhas, e por último colocar as `td` s na `tr` , e por fim a `tr` na tabela. Isto é, há quatro funcionalidades em uma mesma função, o que dificulta a manutenção do código.

Imagine outro desenvolvedor tendo que interpretar o bloco de código gigante, ele terá dificuldade para entender qual a sua utilidade, pois isto não está explícito. O ideal é quebrarmos o código em várias funções menores, o que, além de deixá-lo mais organizado, faz com que as responsabilidades sejam separadas, cada função com uma diferente. Vamos comentar o que cada trecho faz:

```
var botaoAdicionar = document.querySelector("#adicionar-paciente");
botaoAdicionar.addEventListener("click",function(event){
    event.preventDefault();

    var form = document.querySelector("#form-adiciona");
    //Extraindo informacoes do paciente do form
    var nome = form.nome.value;
    var peso = form.peso.value;
    var altura = form.altura.value;
    var gordura = form.gordura.value;
```

Mais abaixo teremos o trecho responsável por criar as tags `tr` e `td` :

```
//cria a tr e a td do paciente
var pacienteTr = document.createElement("tr");

var nomeTd = document.createElement("td");
var pesoTd = document.createElement("td");
var alturaTd = document.createElement("td");
var gorduraTd = document.createElement("td");
var imcTd = document.createElement("td");

nomeTd.textContent = nome;
pesoTd.textContent = peso;
alturaTd.textContent = altura;
gorduraTd.textContent = gordura;
imcTd.textContent = calculaImc(peso,altura);

pacienteTr.appendChild(nomeTd);
pacienteTr.appendChild(pesoTd);
pacienteTr.appendChild(alturaTd);
pacienteTr.appendChild(gorduraTd);
pacienteTr.appendChild(imcTd);
```

Em seguida, temos o trecho responsável por adicionar o paciente.

```
//adicionando o paciente na tabela.
var tabela = document.querySelector("#tabela-pacientes";

tabela.appendChild(pacienteTr);
```

Temos pelo menos três funções que trabalharão com tarefas menores.

Função para capturar os dados do formulário

A primeira coisa que podemos fazer é extrair a responsabilidade de capturar os dados do paciente do formulário para uma nova função que receberá o nome `obtemPacienteDoFormulario()`. Ela receberá o formulário por parâmetro e extrairá os dados dele:

```
var form = document.querySelector("#form-adiciona");
// Extraindo informacoes do paciente do form
obtemPacienteDoFormulario(form)
```

Criaremos a função `obtemPacienteDoFormulario()`, para onde iremos mover as variáveis `nome`, `peso`, `altura` e `gordura`.

```
function obtemPacienteDoFormulario(form) {
    var nome = form.nome.value;
    var peso = form.peso.value;
    var altura = form.altura.value;
    var gordura = form.gordura.value;
}
```

Esse código está pegando todos os valores e extraindo para variáveis. O nome, peso, altura e gordura são características do paciente. Logo, eles pertencem ao mesmo paciente e poderiam ser representados pela mesma coisa. Quando falamos em **representar um paciente**, falamos de **objetos**. Nas linguagens de programação, objetos representam coisas do mundo real, ou mesmo da programação.

Ao criarmos um paciente, sabemos que ele deve ter um nome, peso, altura e gordura. Então, agruparemos todas as características em uma mesma variável criando um objeto em JavaScript usando **chaves** (`{}`):

```
function obtemPacienteDoFormulario(form) {

    var paciente = {

    }

    var nome = form.nome.value;
    var peso = form.peso.value;
    var altura = form.altura.value;
    var gordura = form.gordura.value;
}
```

Dentro das chaves, passamos as **propriedades** do objeto, que nada mais são que as suas características. Para criar uma propriedade, passamos o seu nome e o seu valor, mas não com um igual e sim com dois pontos. Por exemplo, a

propriedade `nome` :

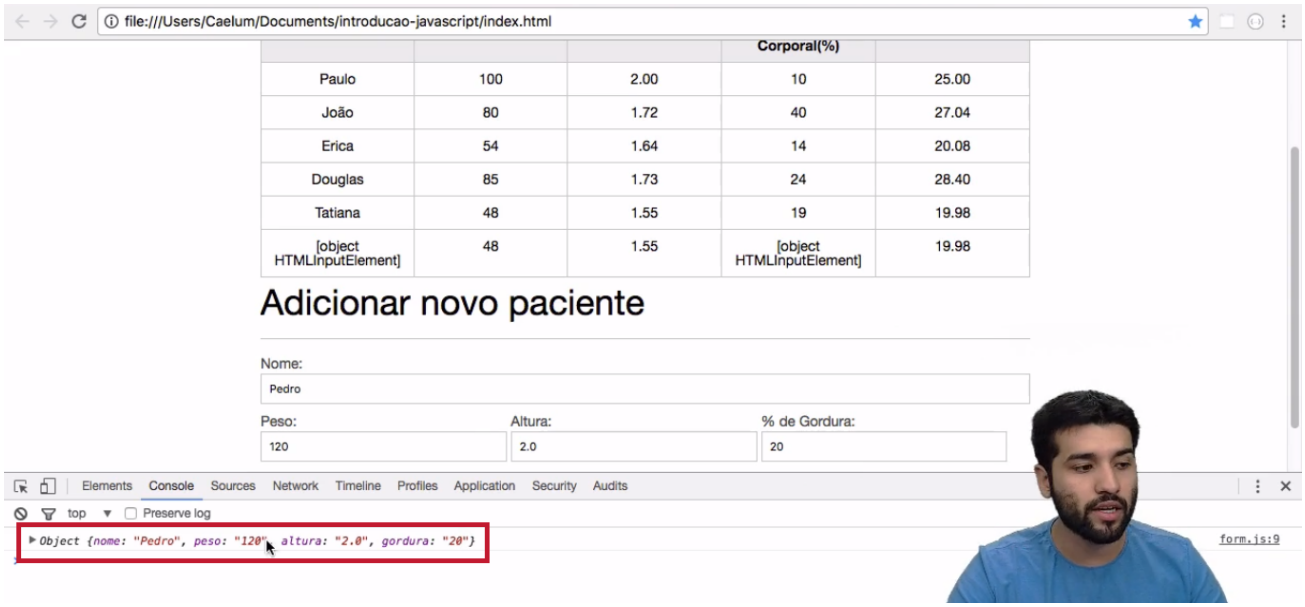
```
function obtemPacienteDoFormulario(form) {

    var paciente = {
        nome: form.nome.value,
        peso: form.peso.value,
        altura: form.altura.value,
        gordura: form.gordura.value
    }
    return paciente;
}
```

Assim, atribuímos às propriedades os valores extraídos do formulário e, no fim, a função retornará o objeto `paciente` .

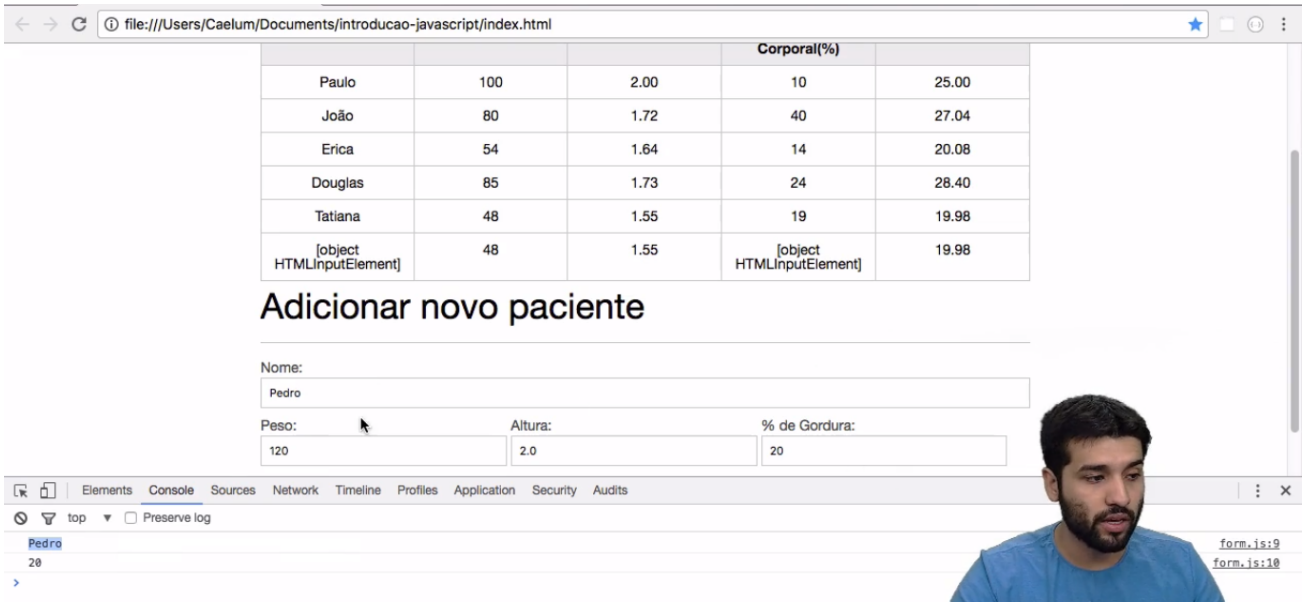
Na parte de cima do arquivo, vamos declarar a variável `paciente` .

```
var botaoAdicionar = document.querySelector("#adicionar-paciente");
botaoAdicionar.addEventListener("click",function(event){
    event.preventDefault();

    var form = document.querySelector("#form-adiciona");
    //Extraindo informacoes do paciente do form
    var paciente = obtemPacienteDoFormulario(form);

    console.log(paciente);
```

Em seguida, testaremos preencher o formulário no browser, após clicarmos no botão "Adicionar", veremos os dados quebrados na ultima linha. No entanto, o objeto será impresso no console.



O objeto `paciente` representa as propriedades do paciente. Se imprimirmos `paciente.nome` , `paciente.gordura` , poderemos acessar cada um dos dados individualmente.



Esta maneira de representar uma variável que contém várias características já foi utilizada anteriormente. Quando selecionamos um elemento com o `querySelector()`, ele também nos devolverá um objeto, como no trecho abaixo do `calcula-imc.js`:

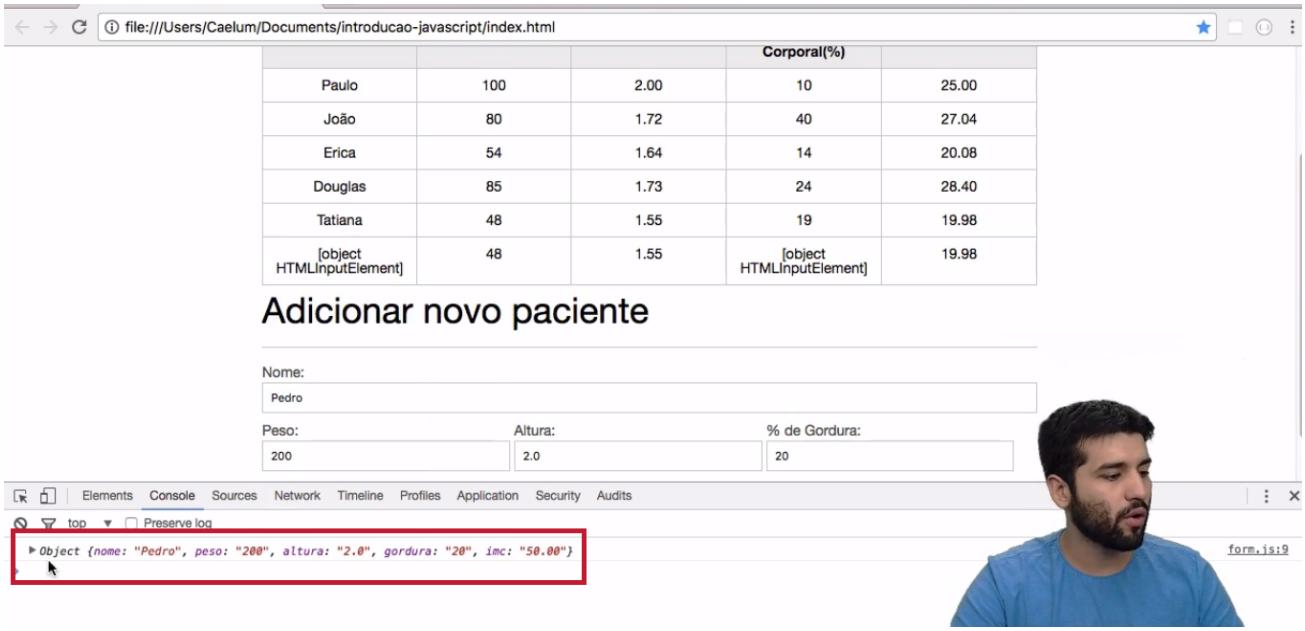
```
var titulo = document.querySelector(".titulo");
titulo.textContent = "Aparecida Nutricionista";
```

No exemplo, o elemento possui a propriedade `textContent` e `value`. Esse trecho do HTML é representado por um objeto. No caso da função `obtemPacienteDoFormulario()` do `form.js`, nós criamos as características do paciente.

Vamos continuar com o código adicionando outra característica: o `imc`. Seu valor será a função criada anteriormente `calculaImc` () e passaremos o peso e a gordura do formulário como parâmetros:

```
function obtemPacienteDoFormulario(form) {

    var paciente = {
        nome: form.nome.value,
        peso: form.peso.value,
        altura: form.altura.value,
        gordura: form.gordura.value,
        imc: calculaImc(form.peso.value, form.altura.value)
    }
}
```

Por fim, a função retornará o objeto que contém todos os dados do paciente, incluindo o `imc`:



Conseguimos representar o paciente adicionado por meio de um objeto e este se encontra disponível no começo do arquivo. Enxugamos as responsabilidades, e podemos continuar com esse processo de melhorias com o restante da função!