Programação Estruturada

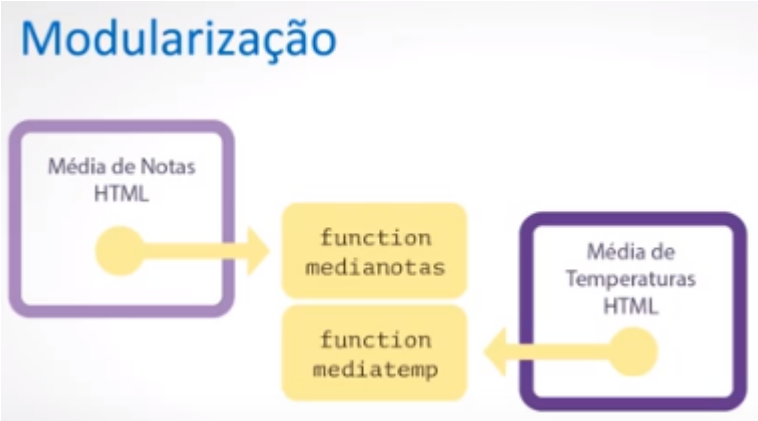
Aula 04 - Funções

Videoaula 03 - Funções em JavaScript (continuação)

Videoaula 03 - Funções em JavaScript (continuação)

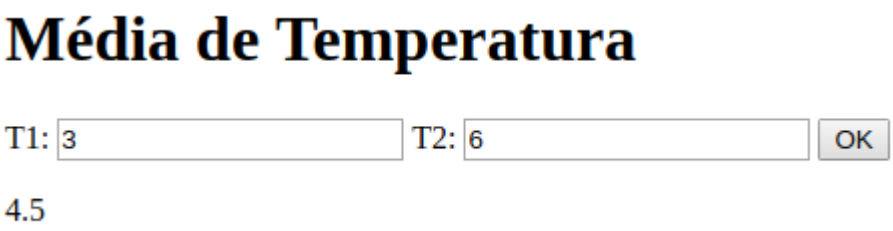
Olá! Na última videoaula, vimos um exemplo de uma página HTML que utilizava uma função para calcular a média das notas de um aluno. Imagine agora que uma outra página HTML também precise calcular a média de dois campos numéricos, porém, em outro contexto onde a página está calculando a média de duas temperaturas (Figura 1). Para ilustrar melhor o problema, vamos supor que os nomes dos campos de entrada e saída são diferentes. O que poderíamos fazer seria, simplesmente, usar uma cópia da função medianotas, que apresentamos, e alterar o nome dos campos de entrada e de saída usados na função, passando a chamar essa nova função de `mediatemp`.

Figura 1 - Modularização



Neste exemplo (Figura 2), temos uma página bastante similar à anterior, mas agora estamos trabalhando com a média de temperaturas. Temos os campos T1 e T2, porém, veja que o comportamento, se eu colocar 3 e 6, vai resultar na mesma média, 4.5.

Figura 2 - Média de Temperaturas



E se olharmos o HTML (Código 1), veremos também que é um HTML bastante parecido com o outro, com dois campos nas linhas 11 e 12, `T1` e `T2`, e eu vou, intencionalmente, alterar o nome desses campos no HTML, o `id` deles, para "`temp1`" e "`temp2`".

Código 1 - 04_3 Média de Temperaturas.html

```
1  <html>
2    <head>
3      <meta charset="UTF-8" />
4      <title>Programação Estruturada - Aula 04</title>
5    </head>
6    <body>
7      <noscript>Seu navegador não suporta JavaScript ou ele está desabilitado.</noscript>
8
9      <h1>Média de Temperatura</h1>
10
11     T1: <input type="number" id="temp1" value="">
12     T2: <input type="number" id="temp2" value="">
13     <button onclick='mediatemp()'>OK</button>
14     <p id="temperatura"></p>
15
16     <script src="script.js"></script>
17
18   </body>
19 </html>
20
```

```
1  function mediatemp() {
2    var x = Number(document.getElementById("temp1").value);
3    var y = Number(document.getElementById("temp2").value);
4
5    document.getElementById("temperatura").innerHTML = (x + y) / 2;
6  }
7
```

Porém, no `onclick`, é chamada a mesma função `mediatemp`, que estará em um outro arquivo JavaScript, chamado “04_3 Media de Temperaturas.js” (Código 2) que, se olharmos para esse arquivo, ele é praticamente igual ao arquivo do último exemplo, onde temos a declaração de duas variáveis, a variável `x` e a variável `y` (linhas 1 e 2).

Código 2 - 04_3 Média de Temperaturas.js

```
1  <html>
2    <head>
3      <meta charset="UTF-8" />
4      <title>Programação Estruturada - Aula 04</title>
5    </head>
6    <body>
7      <noscript>Seu navegador não suporta JavaScript ou ele está desabilitado.</noscript>
8
9      <h1>Média de Temperatura</h1>
10
11     T1: <input type="number" id="temp1" value="">
12     T2: <input type="number" id="temp2" value="">
13     <button onclick='mediatemp()'>OK</button>
14     <p id="temperatura"></p>
15
16     <script src="script.js"></script>
17
18   </body>
19 </html>
20
```

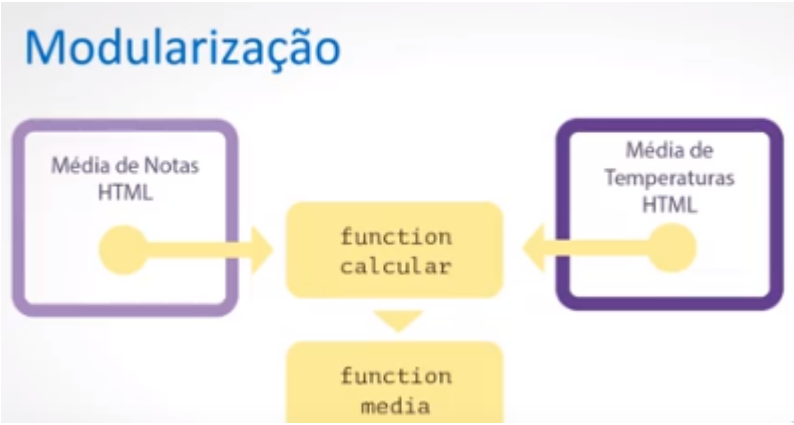
```
1  function mediatemp() {
2    var x = Number(document.getElementById("temp1").value);
3    var y = Number(document.getElementById("temp2").value);
4
5    document.getElementById("temperatura").innerHTML = (x + y) / 2;
6  }
7
```

Nele, a variável `x` recebe o valor do campo "`temp1`", que é exatamente uma das diferenças com relação ao último arquivo. A variável `y` recebe o valor do campo "`temp2`", então alteramos o valor do `innerHTML`, do elemento de saída que tem um nome chamado "`temperatura`", que é outra diferença, porém a fórmula `(x + y) / 2` que nos dá a média de `x` e `y` é a mesma, ou seja, temos uma página com o comportamento bastante similar ao último exemplo.

Você notou que temos duas funções praticamente idênticas que alteram apenas os nomes dos campos de entrada e saída? Ambas as funções usam a mesma fórmula para calcular a média. Na prática, estamos duplicando o código, e isso é muito ruim. Isso porque será necessário testar e manter ambas as funções. Ou seja, caso você precise alterar a fórmula, vai ter de alterar e testar em vários lugares. Além do trabalho de procurar e trocar a fórmula diversas vezes, você corre o risco de esquecer de alterar uma delas e de o programa ficar inconsistente. Para evitar isso, será que não poderíamos “unificar” essas duas funções?

Bem, felizmente, a resposta é sim. A ideia é termos uma função que recebe como parâmetro os nomes dos campos de entrada e de saída, e usa esses nomes para pegar os valores dos campos de entrada e escrever a média no campo de saída. Podemos até mesmo criar uma outra função que isole o cálculo dessa média. Dessa forma, criaremos a função `calcular`, que em seu corpo chamará a função `media` (Figura 3). Isso é muito comum em sistemas grandes, nos quais observamos situações de funções que chamam outras funções.

Figura 3 - Média de Temperaturas

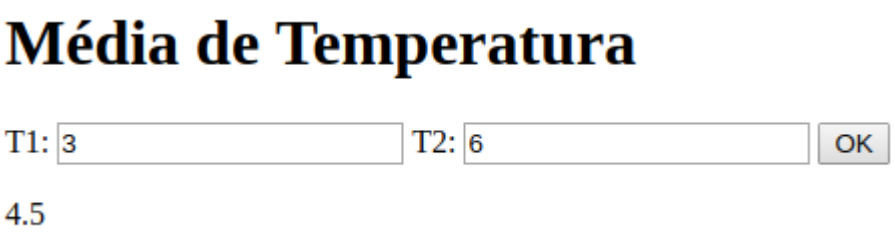


Vamos ver o resultado?

Então, do ponto de vista do comportamento da página, teremos exatamente a mesma coisa que tínhamos nos exemplos anteriores. A página de notas continua a mostrar a média de notas após o clique, e vai escrever a média dos campos que você

colocar. A página das temperaturas também é a mesma coisa (Figura 4), para o valor que você escrever nos campos (T1 e T2) será gerada a média 4.5 para 3 e 6 ou a média 5 para 3 e 7.

Figura 4 - Média de Temperaturas



Porém, a estrutura interna da página é diferente, observe que o HTML de média de notas (Código 3) é muito similar ao que já tínhamos com os dois campos, `N1` e `N2`, porém, no `onclick`, vou chamar uma nova função, `calcular`, que estará nesse arquivo JavaScript "04_4 Media.js" e vai passar os nomes dos campos `N1` e `N2`, que são os campos de entrada, e o nome do campo de saída, que é o campo "`resultado`", que está na linha 14.

Código 3 - 04_4 Média de Notas.html

```
1  <html>
2    <head>
3      <meta charset="UTF-8" />
4      <title>Programação Estruturada - Aula 04</title>
5    </head>
6    <body>
7      <noscript>Seu navegador não suporta JavaScript ou ele está desabilitado.</noscript>
8
9      <h1>Média de Notas (Mais Modularizado)</h1>
10
11     N1: <input type="number" id="N1" value="">
12     N2: <input type="number" id="N2" value="">
13     <button onclick="calcular('N1', 'N2', 'resultado')">OK</button>
14     <p id="resultado"></p>
15
16     <script src="script.js"></script>
17
18   </body>
19 </html>
20
```

```
function calcular(entrada1, entrada2, saida) {
    var x = Number(document.getElementById(entrada1).value);
    var y = Number(document.getElementById(entrada2).value);

    var m = media(x,y);

    document.getElementById(saida).innerHTML = m;
}

function media(a, b) {
    return (a + b)/2;
}

```

Ao acessarmos a página de média de temperaturas (Código 4), veremos que a página vai ser muito similar ao que já vimos nos exemplos anteriores.

Porém, o `onclick` do botão (linha 13) é idêntico ao que vimos na outra página, média de notas (Código 3). Ele também chama a função `calcular`, que estará no mesmo arquivo, ou seja, na prática, os dois arquivos estão importando o mesmo arquivo JavaScript e o `onclick` dos dois botões, nos dois arquivos, estão chamando a mesma função, apenas passando parâmetros diferentes. Assim, no Código 4, a função `calcular` dos campos de entrada passa de `N1` e `N2` (linhas 11 e 12), para '`temp1`' e '`temp2`' e o campo de saída "`temperatura`" (linhas 13 e 14). Mas, na prática, os dois estão chamando a mesma função.

Código 4 - 04_4 Média de Temperaturas.html

```
<html>
    <head>
        <meta charset="UTF-8" />
        <title>Programação Estruturada - Aula 04</title>
    </head>
    <body>
        <noscript>Seu navegador não suporta JavaScript ou ele está desabilitado.</noscript>

        <h1>Média de Temperaturas (Mais Modularizado)</h1>

        N1: <input type="number" id="temp1" value="">
        N2: <input type="number" id="temp2" value="">
        <button onclick="calcular('temp1', 'temp2', 'temperatura')">OK</button>
        <p id="temperatura"></p>

```

```
16      <script src="script.js"></script>
17
18    </body>
19  </html>
20
```

```
1  function calcular(entrada1, entrada2, saida) {
2     var x = Number(document.getElementById(entrada1).value);
3     var y = Number(document.getElementById(entrada2).value);
4
5     var m = media(x,y);
6
7     document.getElementById(saida).innerHTML = m;
8  }
9
10 function media(a, b) {
11    return (a + b)/2;
12 }
13
```

Veja agora essa função no arquivo que os dois HTMLs estão importando, o “04_4 Media.js” (Código 5). Ele tem a função `calcular` que recebe o nome do campo de `entrada1`, o nome do campo de `entrada2` e o nome do campo de `saida` (linha 1).

Código 5 - 04_4 Média.js

```
1  <html>
2     <head>
3        <meta charset="UTF-8" />
4        <title>Programação Estruturada - Aula 04</title>
5     </head>
6     <body>
7        <noscript>Seu navegador não suporta JavaScript ou ele está desabilitado.</noscript>
8
9        <h1>Média de Temperaturas (Mais Modularizado)</h1>
10
11       N1: <input type="number" id="temp1" value="">
12       N2: <input type="number" id="temp2" value="">
13       <button onclick="calcular('temp1', 'temp2', 'temperatura')">OK</button>
14       <p id="temperatura"></p>
15
16       <script src="script.js"></script>
17
18    </body>
19 </html>
20
```

```
1  function calcular(entrada1, entrada2, saida) {
2      var x = Number(document.getElementById(entrada1).value);
3      var y = Number(document.getElementById(entrada2).value);
4
5      var m = media(x,y);
6
7      document.getElementById(saida).innerHTML = m;
8  }
9
10 function media(a, b) {
11     return (a + b)/2;
12 }
13
```

Assim, ele pega o valor do elemento que tem o identificador passado como campo de `entrada1`, atribui a `x` (linha 2), pega o valor do elemento que tem identificador passado como segundo argumento chamado `entrada2`, e atribui a `y` (linha 3). E vai atribuir à variável `m` o resultado da chamada de uma função, a função `media`, passando esses valores que ele pegou como entrada `x` e `y` (linha 5).

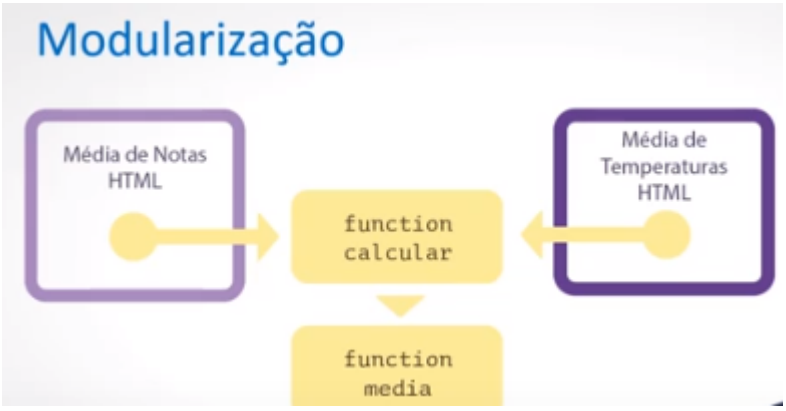
A função `media`, é uma outra função que está entre as linhas 10 e 12, que tem dois parâmetros, `a` e `b` e retorna na linha 11, a média de `a` com `b`. A partir disso, ele soma `a` e `b` e divide por 2, dando o resultado como `retorno`.

Nesse momento, a chamada de função passando `x` e `y` usa esses argumentos e recebe a média de ambos, atribuindo isso à variável `m` (linha 5). Por fim, já visto em aulas anteriores, vamos usar o valor dessa variável para alterar o `innerHTML` (linha 7) do elemento que tem o identificador igual ao que foi utilizado na chamada da função `calcular`, passado como argumento para o parâmetro `saida` (linha 1) e ele vai escrever o valor `m` (linha 7).

Por exemplo, se você acessar o arquivo “04_4 Media de Notas.html” (Código 3), vai ver que passou o identificador ‘`resultado`’ na linha 13, como terceiro argumento. E o que essa função vai fazer é escrever nesse campo chamado de "`resultado`" (linha 14 do Código 3) o valor de `m` (linha 7 do Código 5). Então, é exatamente isso, só que feito de maneira mais modularizada.

Neste exemplo (Figura 5), dizemos que o código de alterar os campos do HTML e o código de calcular a média estão encapsulados dentro das funções `calcular` e `media`, respectivamente, já que o HTML agora chama a função `calcular` para alterar o resultado na página, que, por sua vez, chama a função `media` para calcular a média dos valores obtidos.

Figura 5 - Modularização



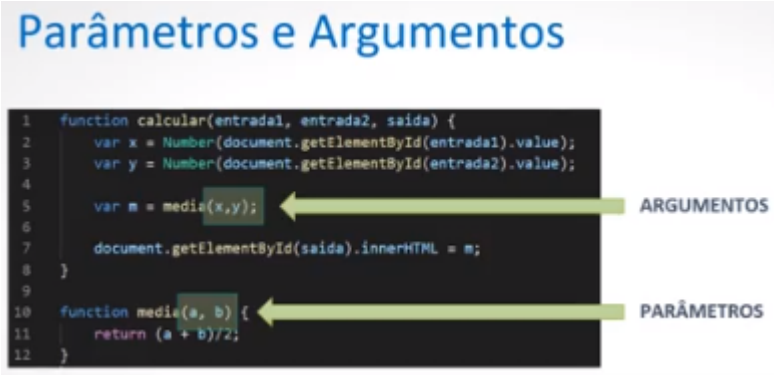
Note que o código HTML não sabe como a função `calcular` está implementada, assim como a própria função `calcular` também não sabe como a função `media` está implementada. Isso permite que você altere os corpos dessas funções sem precisar alterar o corpo do HTML ou da função `calcular`.

No exemplo, ambas as funções tinham parâmetros. Nesse caso, sempre que chamarmos essas funções, precisaremos indicar os valores desses parâmetros. Chamamos esses valores de **argumentos**.

Os termos parâmetro e argumento costumam ser confundidos entre si. No entanto, é importante notar que parâmetro é o nome da variável usada na declaração da função, e argumento é o valor utilizado na chamada da função. Esse valor pode ser um valor constante, uma expressão, por exemplo, aritmética, ou o nome de uma variável, conforme fizemos no exemplo que você pode ver agora no slide (Figura 6).

Neste exemplo, usamos as variáveis `x` e `y` como argumentos para chamar a função `media` que tem parâmetros `a` e `b`.

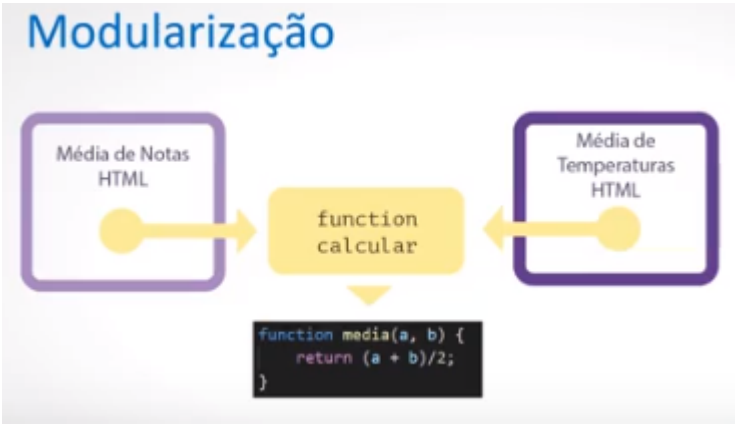
Figura 6 - Parâmetros e Argumentos



As funções podem retornar um valor. Isto, porém, não é obrigatório. Em outras linguagens de programação, esse tipo de função é chamado de procedimento. Ou seja, outras linguagens de programação diferenciam funções, que possuem retorno, de procedimentos, que não possuem retorno.

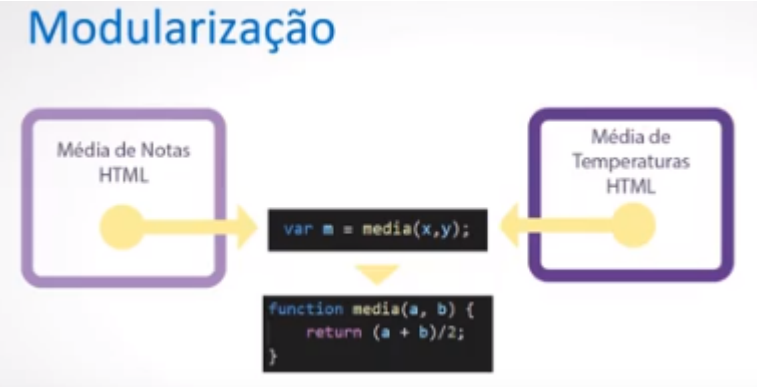
Nesse último exemplo (Figura 7), vimos que a função `media` tem um retorno, o qual é indicado com a palavra-chave `return`. Quando o JavaScript atingir uma declaração de retorno, a função irá parar de executar. Se a função foi chamada a partir de uma instrução, o JavaScript "retornará" para executar o código após a instrução de chamada. O valor de retorno é "devolvido" de volta ao "chamador".

Figura 7 - Função "`media`"



No exemplo (Figura 8), a variável `m` da função `calcular` recebe o valor retornado pela função `media`. Como vimos, esse valor é então usado para atualizar a página HTML.

Figura 8 - Função "`calcular`"



Existe ainda a possibilidade de fazer com que uma função retorne mais do que um valor, utilizando Arrays. Além disso, poderíamos também utilizar Arrays para melhorar ainda mais a modularização do nosso exemplo, fazendo com que a função média recebesse uma quantidade qualquer de campos de entrada e escrevesse no campo de saída a média dos valores de todos os campos. Porém, apresentarei isso em uma outra aula. De qualquer forma, disponibilizo agora para você o código fonte de um exemplo que usa Arrays nesses dois casos.

Código 6 - 04_5 Media de Temperaturas.html

```
<html>
  <head>
    <meta charset="UTF-8" />
    <title>Programação Estruturada - Aula 04</title>
  </head>
  <body>
    <noscript>Seu navegador não suporta JavaScript ou ele está desabilitado.</noscript>

    <h1>Média de Temperaturas (Genérico)</h1>

    N1: <input type="number" id="N1" value="">
    N2: <input type="number" id="N2" value="">
    N3: <input type="number" id="N3" value="">
    <button onclick="calcular(['N1', 'N2', 'N3'], 'resultado')">OK</button>
    <p id="resultado"></p>

    <script src="script.js"></script>
  </body>
</html>

```

```
var total;
```

```
var contador;

function calcular(entradas, saida) {
    total = 0;
    contador = 0;
    entradas.forEach(soma);

    var m = media();

    document.getElementById(saida).innerHTML = m[0] + m[1];
}
function soma(value) {
    total = total + Number(document.getElementById(value).value);
    contador++;
}
function media() {
    return ["A temperatura media é ", total / contador];
}

```

Tudo bem até aqui? Na próxima videoaula, você irá estudar de que forma podemos compor funções em nossos programas.