Programação Estruturada

Aula 04 - Funções

Videoaula 02 - Funções em JavaScript

Videoaula 02 - Funções em JavaScript

Funções

Imagine um programa que calcula a média obtida por um aluno nas suas duas notas de uma disciplina. Esse é um problema pequeno e simples. Poderíamos resolvê-lo usando uma página que tem dois campos numéricos de entrada e alterar o valor de um campo texto de saída com a média dos dois campos de entrada. Veja isso em um exemplo.

Na Figura 1, temos dois campos numéricos, `N1` e `N2`; ao selecionar 3 e 6, respectivamente, e clicar em OK, a média gerada desses dois campos será 4.5, cujo resultado expressa o comportamento esperado.

Figura 1 - Média de Notas sem Modularização

Você pode ver no HTML (Código 1), nas linhas 11 e 12, os dois campos de entrada, o primeiro é o `N1` e o segundo o `N2`, e a declaração do botão (`onclick`) nas linhas 13, 14 e 15.

Código 1 - 04_1 Média de Notas.html

```
1  <html>
2
3    <head>
4      <meta charset="UTF-8" />
5      <title>Programação Estruturada - Aula 04</title>
6    </head>
7
8    <body>
```

```
9       <noscript>Seu navegador não suporta JavaScript ou ele está desabilitado.</noscript>
10
11      <h1>Média de Notas (Sem Modularização)</h1>
12
13      N1: <input type="number" id="N1" value="">
14      N2: <input type="number" id="N2" value="">
15      <button onclick='document.getElementById("resultado").innerHTML =
16                          (Number(document.getElementById("N1").value)
17                          + Number(document.getElementById("N2").value)) / 2'>OK</button>
18
19      <p id="resultado"></p>
20
21    </body>
22
23  </html>
24
```

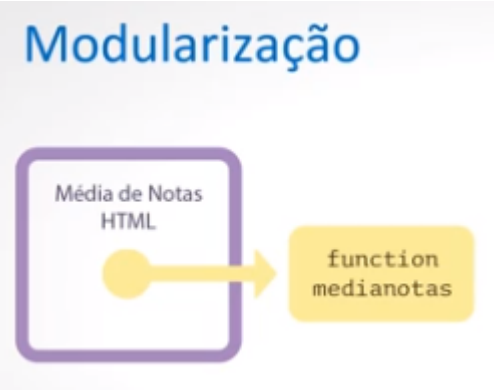
No botão `onclick`, você vê um longo trecho JavaScript (linhas 13 a 15). Nele, o valor do `innerHTML` é alterado no campo "`resultado`", que é o campo texto e que recebe o resultado dos dois valores do `N1` e do `N2`, ou seja, ele pega o valor, na linha 14 do campo `N1`, transforma em número; usa o mesmo comportamento na linha 15, só que dessa vez, no campo `N2`, pega o valor, transforma em número, soma e divide por 2.

Então, esse é o primeiro exemplo de como fazer a média de dois campos de entrada.

Você notou como fica difícil entender a página HTML quando escrevemos o código dessa maneira? Imagine como seria difícil manter todo um sistema se estivéssemos usando essa forma de escrever a página. Felizmente, já vimos como melhorar o entendimento para esse caso.

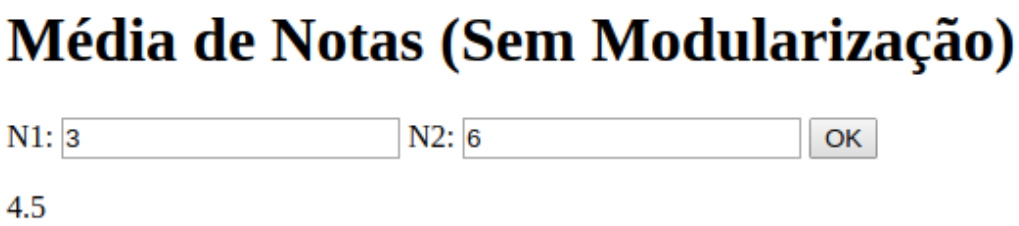
Mais precisamente, podemos escrever a alteração do campo texto em uma função JavaScript (Figura 2) que pode ser chamada, por exemplo, `medianotas`. Para facilitar ainda mais a leitura do código, usaremos variáveis para armazenar separadamente o valor de cada um dos campos de entrada. Além disso, para melhorar a estrutura de nossos arquivos, escreveremos o código JavaScript em um arquivo externo.

Figura 2 - Código JavaScript em um arquivo externo



Veja só! Nesse segundo exemplo, em relação à página HTML, o comportamento é praticamente o mesmo: se eu colocar uma nota (3 e 6, por exemplo) e clicar em OK, novamente a média vai aparecer, ou seja, o resultado será 4.5 (Figura 3).

Figura 3 - Média de Notas com Modularização



Porém, se olharmos o HTML (Código 2), veremos uma sessão JavaScript (linha 16) que vai referenciar um `source`, ou seja, um arquivo JavaScript externo, e, na linha 13, o `onclick` invoca a função `medianotas` deste arquivo. Essa função estará declarada exatamente no arquivo "04_2 Media de Notas.js" (Código 3), que é o arquivo JavaScript.

Código 2 - 04_2 Média de Notas.html

```
1  <html>
2
3    <head>
4      <meta charset="UTF-8" />
5      <title>Programação Estruturada - Aula 04</title>
6    </head>
7
8    <body>
9      <noscript>Seu navegador não suporta JavaScript ou ele está desabilitado.</noscript>
10
```

```
    <h1>Média de Notas (Com Modularização)</h1>

    N1: <input type="number" id="N1" value="">
    N2: <input type="number" id="N2" value="">
    <button onclick='medianotas()'>OK</button>
    <p id="resultado"></p>

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

  </body>

</html>

```

```
function medianotas() {
  var x = Number(document.getElementById("N1").value);
  var y = Number(document.getElementById("N2").value);

  document.getElementById("resultado").innerHTML = (x + y) / 2;
}

```

Nesse arquivo, entre as linhas 1 e 6, foi declarada a function `medianotas`; na linha 2, pegamos o valor do elemento `N1` da página, atribuindo isso à variável `x`. Na linha 3, fizemos a mesma coisa, só que para o campo `N2`, e atribuímos isso à variável `y`. E, na linha 5, alteramos o valor do `innerHTML` do campo "`resultado`" para a média de `x` e `y`, ou seja, somamos `x` com `y` e dividimos por 2.

Código 3 - 04_2 Média de Notas.js

```
<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 Notas (Com Modularização)</h1>

    N1: <input type="number" id="N1" value="">
    N2: <input type="number" id="N2" value="">
    <button onclick='medianotas()'>OK</button>
```

```
    <p id="resultado"></p>

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

  </body>

</html>

```

```
function medianotas() {
  var x = Number(document.getElementById("N1").value);
  var y = Number(document.getElementById("N2").value);

  document.getElementById("resultado").innerHTML = (x + y) / 2;
}

```

Como eu mostrei, o comportamento é exatamente o mesmo da outra página, só que agora foi feito algo mais modularizado (Figura 3).

Agora, você deve ter notado que ficou bem mais fácil de entender o código, não é verdade?

Estrutura das Declarações de Funções

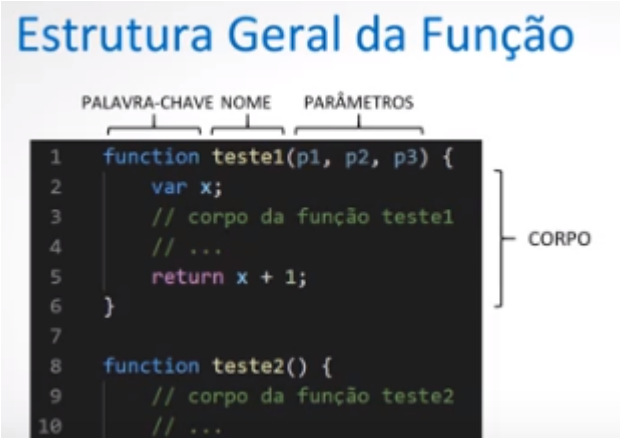
Bem, de uma maneira geral, uma função JavaScript é definida usando a palavra-chave `function`, seguida por um nome e parênteses. As funções podem, opcionalmente, receber valores, que chamamos de **parâmetros**, e podem, também opcionalmente, retornar um valor, o qual chamamos de **retorno da função**.

As regras para os nomes das funções são as mesmas que temos para os nomes das variáveis, ou seja:

- devem começar com uma letra, $ ou _
- podem conter apenas letras, dígitos, sublinhados e $
- não podem ser palavras reservadas JavaScript

No slide (Figura 4), temos a estrutura da definição de duas funções, `teste1` e `teste2`.

Figura 4 - Estrutura Geral da Função

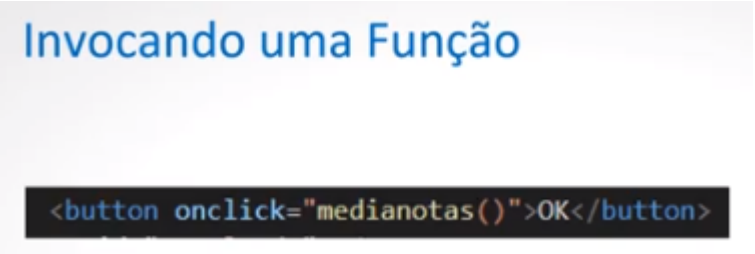


Os parâmetros da função, ou seja, os valores que a função pode receber como entrada, têm seus nomes listados dentro dos parênteses. Caso tenhamos mais de um parâmetro, como na função `teste1` do slide, os nomes dos parâmetros são separados por vírgulas. Caso não haja parâmetros, como na função `teste2` do slide, temos apenas os parênteses sem nada entre eles.

Por fim, temos o código a ser executado pela função, também chamado de **corpo da função**, colocado dentro das chaves, abrindo e fechando o bloco da função. Nesse bloco, ou seja, no corpo da função, os parâmetros funcionam como variáveis locais. Além disso, no corpo da função, podemos declarar variáveis (chamadas de variáveis locais) e comandos da linguagem. Uma variável local somente pode ter seu valor acessado dentro da função em que foi declarada. No nosso exemplo, as variáveis locais `p1, p2, p3` e `x` só podem ser referenciadas dentro do corpo da função `teste1` e na expressão de retorno dessa função.

O código dentro da função será executado quando "algo" invocar, ou, como prefiro dizer, chamar, a função. Isso pode acontecer quando um evento ocorre no HTML. No nosso exemplo (Figura 5), vimos que quando um usuário clica no botão OK, a função `medianotas` é chamada e seu corpo executado. A chamada de uma função também pode acontecer dentro do próprio código JavaScript. De fato, uma função pode ser chamada quantas vezes forem necessárias dentro de um arquivo JavaScript.

Figura 5 - Invocando uma Função



Voltando ao nosso exemplo, você sabia que podemos melhorar ainda mais a modularização dele? Na próxima videoaula, você verá como fazer isso e aprenderá muito mais sobre chamada de funções. Até lá!