UNINOVE

# Funções

INTRODUÇÃO A CRIAÇÃO E MANIPULAÇÃO DE FUNÇÕES EM JAVASCRIPT

AUTOR(A): PROF. DANIEL FERREIRA DE BARROS JUNIOR

## Funções

Criar um código em JavaScript bem estruturado e reutilizável é desejado e podemos considerar como uma boa prática de programação. Neste sentido, a utilização de funções tem um papel importante, pois auxiliar não somente na organização de seu código fonte, bem como na estruturação de um código reutilizável e ajustável.
A estrutura básica de uma função se dá conforme o modelo a seguir:

```
1. <script type="text/javascript">
2.     function nomeFuncao(<valores externos>){
3.         <comandos>
4.         <comandos>
5.         .....
6.         <comandos>
7.
8.         return <valor a retornar>;
9.     }
10. </script>
```

Alguns elementos são fundamentais e não devem faltar:

- Nome da função
- Valores externos (argumentos)
- Retorno

Ao nome da função deve-se seguir as mesmas regras de nomenclatura das variáveis. Esta nomenclatura deve ser precedida pela palavra function, que determina o bloco de comandos de uma função. A seguir, em parênteses, deve-se obter os valores externos ou argumentos a esta função. Caso seja necessário receber

mais de um valor, basta separar os elementos por virgula.

Observe que a função é delimitada pelo uso de chaves { } .Entre estas chaves inserimos nosso código fonte em JavaScritp.

Sobre o retorno, ou return, este comando determina qual resultado será retornado, ou a saída desta função. Neste caso, toda função deve ter um retorno informado.

Outro ponto importante é a chamada desta função.

Para fazer uso de uma função, em algum momento do código JavaScript, deve haver uma chamada de função. O uso mais comum atribuir esta chamada a uma variável, uma vez que as funções devem retornar valor. Nesta "chamada" de função, deve-se caso necessário, já informar quais argumentos serão "passados" a função solicitada.

Vamos analisar o código a seguir:

```
1. <script type="text/javascript">
2.     function dobro(x){
3.         return x*2;
4.     }
5.
6.     var n = dobro(4);
7.     document.write(n);
8.     // Resultado: 8
9. </script>
```

Neste código temos uma função chamada dobro, que recebe um argumento x, e retorna o dobro do valor informado na função. Na sequência temos uma variável n inicializada com o valor retornada da função dobro. Observe que na mesma linha é informada o argumento da função.

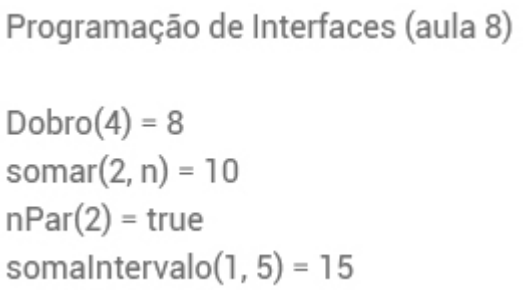
Quando o comando document.write(n) é executado ele informa o valor 8, pois é o dobro do valor de argumento passado a função dobro.

Vamos analisar um novo código:

```
<!DOCTYPE html>
<html>
    <head>
        <title>Tópico 7</title>
        <meta charset="UTF-8">
    </head>
    <body>
        <div>Programação de Interfaces (aula 7)</div>
        <script type="text/javascript">
            function dobro(x){
                return x*2;
            }

            function somar(a,b){
                return a+b;
            }

            function nPar(n){
                if (n%2==0)
                    return true;
                else
                    return false;
            }

            function somaIntervalo(x, y){
                var soma=0;
                for(var i=x; i<=y; i++){
                    soma += i;
                }
                return soma;
            }

            var n = dobro(4);
            document.write("<br>Dobro(4) = " + n);

            var soma = somar(2,n);
            document.write("<br>somar(2, n) = ", soma);

            document.write("<br>nPar(2) = ", nPar(2));

```

```
41.             var somatoria = somaIntervalo(1,5);
42.             document.write("<br>somaIntervalo(1, 5) = " + somatoria);
43.
44.         </script>
45.     </body>
46. </html>
```

Para este código JavaScript temos o seguinte resultado:




Vamos comentar os principais pontos.

A linha 14, a function somar(a,b) recebe dois parâmetros, ou argumentos. Observe que estes devem estar separados por virgula, e se caso fosse necessário mais valores, bastaria adicionar novos argumentos separados por virgula.

Na linha 18, a function nPar(n), recebe um valor que é testado se o mesmo é par. O retorno é dado como true ou false, ou seja, valores booleanos.

Na linha 25, a function somaIntervalo(x,y) recebe dois parâmetros. O primeiro determinando o início de uma sequência numérica, e o segundo argumento determinando o fim deste intervalo. Observe o uso de uma estrutura de repetição dentro desta function.

Na linha 36, a variável soma recebe o valor de retorno da função somar(2, n). Entenda que o valor de n já estava determinada anteriormente, na linha 33, ou seja, a passagem de argumentos pode ser valores literais ou de variáveis.

Na linha 39, escrevemos o resultado da function nPar(2). Verifique que não determinamos o retorno desta função a uma outra variável, ou seja, é possível exibir o resultado de uma function sem o seu armazenamento prévio em outra variável auxiliar. Outro ponto importante é que a saída desta função é dada como true ou false.

A linha 41, a function somaIntervalo(1,5), envia o seu retorno a variável somatória. Observe que independentemente da quantidade de trabalho ou codificação em uma função, a sua utilização e retorno é simples. Esta característica auxilia a simplificar e organizar nosso código JavaScript.

## DICA:

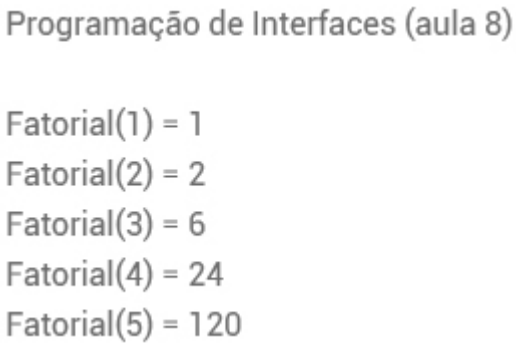
> Observe com atenção! As variáveis criadas “dentro” de uma função só podem ser acessadas dentro da mesma, ou seja, não podemos chamar uma variável de fora da função, apenas o seu retorno. Isso permite reutiliza uma mesma nomenclatura de variável, desde que estejam protegidas em funções diferentes.

# Função Recursiva

Um recurso importante é o uso da recursividade, ou seja, o uso de uma função chamando a si mesma para obter o resultado desejado.
Vamos analisar o exemplo a seguir:

```
1. <!DOCTYPE html>
2. <html>
3.     <head>
4.         <title>Tópico 7</title>
5.         <meta charset="UTF-8">
6.     </head>
7.     <body>
8.         <div>Programação de Interfaces (aula 8)</div>
9.         <script type="text/javascript">
10.             function fatorial(n){
11.                 if ((n == 0) || (n == 1))
12.                     return 1;
13.                 else
14.                     return (n * fatorial(n - 1));
15.             }
16.
17.             document.write("<br>Fatorial(1) = " + fatorial(1));
18.             document.write("<br>Fatorial(2) = " + fatorial(2));
19.             document.write("<br>Fatorial(3) = " + fatorial(3));
20.             document.write("<br>Fatorial(4) = " + fatorial(4));
21.             document.write("<br>Fatorial(5) = " + fatorial(5));
22.
23.         </script>
24.     </body>
25. </html>
```

A este código JavaScript temos o seguinte resultado:




Observe que em um dado momento, na linha 14, o retorno da function fatorial(n) calcula um valor e chama a si mesma para obtenção do seu resultado.
No caso do fatorial(4), a function fatorial está executando a operação matemática: 4*3*2*1, tendo como resultado o valor 24.

## DICA:

Para saber mais, veja em:
https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Guide/Fun%C3%A7%C3%B5es (https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Guide/Fun%C3%A7%C3%B5es)
https://msdn.microsoft.com/pt-br/library/wwbyhkx4(v=vs.94).aspx (https://msdn.microsoft.com/pt-br/library/wwbyhkx4(v=vs.94).aspx)

## RESUMO:

Neste tópico observamos:

- Criação e utilização de funções.
- Exemplos sobre funções.
- Funções Recursivas.

# Conclusão

A utilização de funções é uma ação e prática recomendada sempre que possível. Sua disponibilidade permite uma melhor divisão entre os objetivos do programa, bem como sua reutilização e melhor manutenção de código.

O uso de recursão é um poderoso recurso se bem utilizado. Mais utilize com cuidado, pois sua verificação e análise não é trivial.

# ATIVIDADE

Escolha a alternativa que indica o valor que será exibido ao executar o script abaixo:?

```
<script type="text/javascript">
   function calcular(x){
      return (x*2);
   };
   function somar(n){
      return n + calcular(n);
   };
   document.write("<br>Valor = " + somar(2));
</script>
```

A. 2
B. 4
C. 6
D. 8

# ATIVIDADE

Escolha a alternativa que indica o valor que será exibido ao executar o script abaixo:?

```
<script type="text/javascript">
   function calcular(x){
      return (x*2);
   };
```

```
    function somar(n){
        return n + calcular(n);
    };
    document.write("<br>Valor = " + somar(2));
</script>
```

A. 2

B. 4

C. 6

D. 8

## ATIVIDADE

Para o resultado 4, selecione a alternativa correta dado o script a seguir:

```
function somar(a,b){
        return a+b;
}
```

A. somar(2,2);

B. somar(2:2);

C. somar(2;2);

D. somar[2,2];

## ATIVIDADE

Escolha a alternativa que indica o valor que será exibido ao executar o script abaixo:?

```
<script type="text/javascript">
    function dobro(x){
        return x*2;
```

```
};
document.write("<br>Valor = " + dobro(dobro(2)) );
</script>
```

A. Valor = 4

B. Valor = 16

C. Valor = 12

D. Valor = 8

# REFERÊNCIA


MORRISON, M. Use a cabeça JavaScript. 5º Ed. Rio de Janeiro: Alta Books, 2012. 606 p.

OLIVIERO. C. A. J. Faça um site JavaScript orientado por projeto. 6º ed. São Paulo: Érica, 2010. 266 p.

ZAKAS, Nicholas C. JavaScript de alto desempenho. 8º Ed. São Paulo: Novatec, 2010. 245 p.