

Lógica de programação com JavaScript

Explicação e tutorial passo a passo para os comandos, em javascript

Sumário:

Sumário:	1
Introdução	2
Manipulando Elementos HTML com JavaScript	2
Exemplo 1: Capturando o valor de um Campo de Texto (value).....	2
Exemplo 2: Capturando o valor de um Campo Numérico (value).....	3
Exemplo 3: Alterando o Valor de um Campo de Texto (value).....	3
Exemplo 4: Alterando um Texto na Página (innerText).....	3
Estruturas Condicionais (If e Switch Case)	4
Condição com if, else if e else.....	4
Exemplo Prático: Verificação de Notas.....	4
Algoritmo (Passo a Passo).....	4
Pseudocódigo.....	4
Código em JavaScript usando if.....	5
Condição com switch case.....	6
Exemplo Prático: Verificação do dia da semana.....	6
Algoritmo (Passo a Passo).....	6
Pseudocódigo.....	7
Código em JavaScript usando switch.....	8
Laços de Repetição (for, while, do while)	9
Laço for.....	9
Laço while.....	9
Laço do while.....	9
Exemplo Prático: Tabuada.....	10
Algoritmo (Passo a Passo).....	10
Usando PARA (for).....	11
Usando ENQUANTO (while).....	12
Usando REPITA (do while).....	13
Declaração de Variáveis	14
Exemplo de var e let no escopo de bloco (if, case, for, while, do-while):.....	14
Exemplo de var, no escopo de função (function):.....	15

Introdução

Lógica de programação é a base para desenvolver qualquer software. Ela envolve o uso de sequências lógicas para resolver problemas, criando algoritmos que descrevem passo a passo como um programa deve funcionar.

O JavaScript é uma das linguagens mais populares para programação, sendo amplamente utilizado para desenvolver páginas web interativas. Ele permite manipular elementos HTML, processar dados e criar aplicações dinâmicas.

Para programar bem, é essencial compreender a lógica por trás das instruções que damos ao computador. Isso inclui conceitos como variáveis, operadores, estruturas condicionais e laços de repetição.

Manipulando Elementos HTML com JavaScript

O JavaScript permite manipular elementos HTML dinamicamente, tornando as páginas interativas. Isso pode ser feito alterando textos, valores de inputs, cores, visibilidade e outros atributos.

O acesso ao documento HTML é feito utilizando o objeto “**Document**”, que representa todo o conteúdo da página carregada, enquanto para acessar um objeto específico através do Id atribuído a ele, utiliza-se a função “**getElementById(<id do elemento>)**”, sendo que “**<id do elemento>**” refere-se ao mesmo id utilizado na declaração da tag html.

Duas propriedades essenciais para essa manipulação são:

1. **value** – Usado para acessar ou modificar o valor de campos de formulário (**input**, **textarea**).
2. **innerText** – Usado para modificar apenas o texto visível dentro de um elemento (**p**, **h1**, **div**, etc.).

Vamos ver cada um desses conceitos com exemplos simples!

Exemplo 1: Capturando o valor de um Campo de Texto (**value**)

html:

```
<input type="text" id="campoTexto">
<button onclick="LerValor()">Ler Valor</button>
```

javascript:

```
var valor = "";
function LerValor() {
    valor = document.getElementById("campoTexto").value;
}
```

Neste exemplo, ao clicar no botão, o valor do campo de texto será inserido na variável “**valor**”

Exemplo 2: Capturando o valor de um Campo Numérico (**value**)

html:

```
<input type="number" id="campoNumerico">
<button onclick="LerValor()">Ler Valor</button>
```

javascript:

```
var valor = 0;
function LerValor() {
    valor = Number(document.getElementById("campoNumerico").value);
}
```

IMPORTANTE: Utilize a função interna “Number” para converter o texto do Input para um número.

Neste exemplo, ao clicar no botão, o valor do campo **numérico** será inserido na variável “**valor**”. Note que neste cenário, é necessário utilizar a função interna “Number” para converter o texto do “input” para número.

Exemplo 3: Alterando o Valor de um Campo de Texto (**value**)

html:

```
<input type="text" id="campoTexto">
<button onclick="alterarValor()">Alterar Valor</button>
```

javascript:

```
function alterarValor() {
    document.getElementById("campoTexto").value = "Novo valor!";
}
```

Neste exemplo, ao clicar no botão, o valor dentro do campo de texto será alterado para “**Novo valor!**”.

Exemplo 4: Alterando um Texto na Página (**innerText**)

html:

```
<p id="meuParagrafo">Texto original.</p>
<button onclick="mudarTexto()">Mudar Texto</button>
```

javascript:

```
function mudarTexto() {
    document.getElementById("meuParagrafo").innerText = "Texto alterado!";
}
```

Aqui, o texto do parágrafo será substituído por “**Texto alterado!**” quando o botão for pressionado.

Estruturas Condicionais (If e Switch Case)

As **estruturas condicionais** são usadas para tomar decisões no código. Dependendo de uma condição, o programa pode seguir caminhos diferentes.

Condição com **if**, **else if** e **else**

A estrutura **if** verifica se uma condição é verdadeira e executa um bloco de código. Se for falsa, podemos usar **else** para definir uma ação alternativa.

javascript:

```
if (condição) {  
    // Código se a condição for verdadeira  
} else {  
    // Código se a condição for falsa  
}
```

Também podemos usar **else if** para testar várias condições.

Exemplo Prático: Verificação de Notas

Vamos criar um algoritmo para verificar se um aluno foi aprovado, ficou de recuperação ou foi reprovado com base na sua nota.

Algoritmo (Passo a Passo)

1. Leia a nota do aluno.
2. Se a nota for maior ou igual a 7, ele está **aprovado**.
3. Se a nota estiver entre 5 e 6.9, ele está de **recuperação**.
4. Se a nota for menor que 5, ele está **reprovado**.
5. Exiba o resultado.

Pseudocódigo

```
INÍCIO  
    ESCREVA "Digite a nota do aluno:"  
    LEIA nota  
    SE nota >= 7 ENTÃO  
        ESCREVA "Aprovado!"  
    SENÃO SE nota >= 5 E nota < 7 ENTÃO  
        ESCREVA "Em recuperação."  
    SENÃO  
        ESCREVA "Reprovado."  
    FIM SE  
FIM
```

Código em JavaScript usando if

```
// Leia a nota do aluno
let nota = document.getElementById("nota").value;

let resultado;

// Se a nota for maior ou igual a 7, ele está aprovado
if (nota >= 7) {
    resultado = "Aprovado!"

// Senão, se a nota estiver entre 5 e 6.9, ele está em recuperação
} else if (nota >= 5 && nota < 7) {
    resultado = "Em recuperação"

// Senão, ele está reprovado
} else {
    resultado = "Reprovado."
}

//Exiba o resultado
document.getElementById("situacao").innerText = resultado;
```

Condição com `switch case`

Quando temos várias possibilidades para uma mesma variável, podemos usar `switch`, que funciona como um menu de escolhas.

javascript:

```
switch (variavel) {  
  case valor1:  
    // Código se a variável for igual a valor1  
    break;  
  case valor2:  
    // Código se a variável for igual a valor2  
    break;  
  default:  
    // Código se nenhum dos casos anteriores for atendido  
}
```

O `break` serve para interromper a execução do `switch` depois que um caso for encontrado.

Exemplo Prático: Verificação do dia da semana

Vamos criar um sistema que, ao informar o número de um dia (de 1 a 7), retorna o nome do dia da semana.

Algoritmo (Passo a Passo)

1. Leia o número do dia da semana (1 a 7).
 2. Verifique o dia da semana usando o `switch`.
 - Se o número for 1, é Segunda-feira.
 - Se for 2, é Terça-feira.
 - Se for 3, é Quarta-feira.
 - Se for 4, é Quinta-feira.
 - Se for 5, é Sexta-feira.
 - Se for 6, é Sábado.
 - Se for 7, é Domingo.
 3. Exiba o nome do dia da semana correspondente.
-

Pseudocódigo

INÍCIO

ESCREVA "Digite o número do dia da semana (1 a 7):"

LEIA dia

OBTENHA o nome do dia usando SWITCH:

CASO dia == 1:

ESCREVA "Segunda-feira"

CASO dia == 2:

ESCREVA "Terça-feira"

CASO dia == 3:

ESCREVA "Quarta-feira"

CASO dia == 4:

ESCREVA "Quinta-feira"

CASO dia == 5:

ESCREVA "Sexta-feira"

CASO dia == 6:

ESCREVA "Sábado"

CASO dia == 7:

ESCREVA "Domingo"

CASO CONTRÁRIO:

ESCREVA "Número inválido, por favor insira um valor entre 1 e 7."

FIM

Código em JavaScript usando switch

```
// Leia o número do dia da semana
let diaDaSemana = Number(document.getElementById("dia").value);
let nomeDoDia;

// Verifica o dia da semana usando switch
switch (diaDaSemana) {
    case 1:
        nomeDoDia = "Domingo";
        break;
    case 2:
        nomeDoDia = "Segunda";
        break;
    case 3:
        nomeDoDia = "Terça";
        break;
    case 4:
        nomeDoDia = "Quarta";
        break;
    case 5:
        nomeDoDia = "Quinta";
        break;
    case 6:
        nomeDoDia = "Sexta";
        break;
    case 7:
        nomeDoDia = "Sábado";
        break;
    default:
        nomeDoDia = "Número inválido, por favor insira um valor entre 1 e 7.";
        break;
}

// Exiba o resultado
document.getElementById("resultado").innerText = nomeDoDia;
```


Laços de Repetição (for, while, do while)

Os laços de repetição são usados para executar um bloco de código **várias vezes** sem precisar repetir manualmente as mesmas instruções.

Laço for

O **for** é utilizado quando sabemos **quantas vezes** a repetição deve ocorrer.

```
for (inicialização; condição; incremento) {  
    // Código a ser repetido  
}
```

Laço while

O **while** repete um bloco de código **enquanto** uma condição for verdadeira. Ele é útil quando não sabemos exatamente quantas vezes a repetição acontecerá.

```
while (condição) {  
    // Código a ser repetido  
}
```

Laço do while

O **do while** funciona como o **while**, mas garante que o código será executado pelo **menos uma vez**, pois a verificação da condição ocorre **depois** da primeira execução.

```
do {  
    // Código a ser repetido  
} while (condição);
```

Exemplo Prático: Tabuada

O objetivo é gerar a tabuada de um número escolhido, multiplicando de 1 a 10.

Algoritmo (Passo a Passo)

1. Leia um número para gerar a tabuada.
2. Crie um laço que multiplica esse número de 1 a 10.
3. Exiba a tabuada.

Usando PARA (for)

pseudocódigo:

```
INÍCIO
  DECLARE numero, i COMO INTEIRO
  DECLARE tabuada COMO CADEIA

  LEIA(numero)
  tabuada = "Tabuada do " + numero + ":\n"

  PARA i = 1 ATÉ 10 FAÇA
    tabuada += tabuada + numero
    tabuada += tabuada + " x "
    tabuada += tabuada + i
    tabuada += tabuada + " = "
    tabuada += tabuada + (numero * i)
    tabuada += tabuada + "\n"
  FIM PARA

  ESCREVA tabuada
FIM
```

javascript:

```
// Leia um número para gerar a tabuada
let numero = document.getElementById("numero").value;

let tabuada = "Tabuada do " + numero + ":\n";

// Crie um laço que multiplica esse número de 1 a 10
for (let i = 1; i <= 10; i++) {
  tabuada += numero;
  tabuada += " x ";
  tabuada += i;
  tabuada += " = ";
  tabuada += (numero * i);
  tabuada += "\n";
}

// Exiba a tabuada
document.getElementById("tabuada").innerText = tabuada;
```

Usando ENQUANTO (while)

pseudocódigo:

```
INÍCIO
  DECLARE numero, i COMO INTEIRO
  DECLARE tabuada COMO CADEIA

  LEIA(numero)
  i = 1
  tabuada = "Tabuada do " + numero + ":\n"

  ENQUANTO i <= 10 FAÇA
    tabuada += tabuada + numero
    tabuada += tabuada + " x "
    tabuada += tabuada + i
    tabuada += tabuada + " = "
    tabuada += tabuada + (numero * i)
    tabuada += tabuada + "\n"
    i = i + 1
  FIM ENQUANTO

  ESCREVA tabuada
FIM
```

javascript:

```
// Leia um número para gerar a tabuada
let numero = document.getElementById("numero").value;

let tabuada = "Tabuada do " + numero + ":\n";

// Crie um laço que multiplica esse número de 1 a 10
let i = 1;
while (i <= 10) {
  tabuada += numero;
  tabuada += " x ";
  tabuada += i;
  tabuada += " = ";
  tabuada += (numero * i);
  tabuada += "\n";
  i++;
}

// Exiba a tabuada
document.getElementById("tabuada").innerText = tabuada;
```

Usando REPITA (do while)

pseudocódigo:

```
INÍCIO
  DECLARE numero, i COMO INTEIRO
  DECLARE tabuada COMO CADEIA

  LEIA(numero)
  i = 1
  tabuada = "Tabuada do " + numero + ":\n"

  REPITA
    tabuada += tabuada + numero
    tabuada += tabuada + " x "
    tabuada += tabuada + i
    tabuada += tabuada + " = "
    tabuada += tabuada + (numero * i)
    tabuada += tabuada + "\n"
    i = i + 1
  ATÉ QUE i <= 10

  ESCREVA tabuada
FIM
```

javascript:

```
// Leia um número para gerar a tabuada
let numero = document.getElementById("numero").value;

let tabuada = "Tabuada do " + numero + ":\n";

// Crie um laço que multiplica esse número de 1 a 10
let i = 1;
do {
  tabuada += numero;
  tabuada += " x ";
  tabuada += i;
  tabuada += " = ";
  tabuada += (numero * i);
  tabuada += "\n";
  i++;
} while (i <= 10);

// Exiba a tabuada
document.getElementById("tabuada").innerText = tabuada;
```

Declaração de Variáveis

Em JavaScript, as variáveis podem ser declaradas de duas maneiras principais: **var** e **let**

- **var**: Tem escopo global ou de função. Isso significa que, se você declarar uma variável com **var** fora de uma função, ela será global. Se você declarar dentro de uma função, ela será acessível dentro dessa função. Uma característica importante do **var** é que ele não respeita o escopo de bloco, ou seja, a variável declarada com var estará disponível fora de estruturas como if, for, entre outros, desde que estejam dentro do mesmo escopo de função ou global.
- **let**: Tem escopo de bloco. Ou seja, uma variável declarada com **let** dentro de um bloco de código (como um if, for, while etc.) só será acessível dentro desse bloco

Exemplo de var e let no escopo de bloco (if, case, for, while, do-while):

```
// INÍCIO DO BLOCO "IF" -----
if (2 > 1) {

    // Declaração de uma variável "LET" dentro do BLOCO
    let nome = "Alexandre";

    // Declaração de variável "VAR" dentro do BLOCO
    var idade = 33;

    // Funciona, apesar de ser um "LET", está dentro do BLOCO em que foi declarado
    console.log(nome); // Escreva(nome)

    // Funciona dentro e fora do BLOCO , por ser um "VAR"
    console.log(idade); // Escreva(idade)
}
// FIM DO BLOCO "IF" -----

// ERRO, pois é um "LET" que foi declarado dentro do BLOCO "if", mas está sendo chamado fora
console.log(nome); // Escreva(nome)

// FUNCIONA, pois apesar de ter sido declarado dentro do BLOCO, é um "VAR"
console.log(idade); // Escreva(idade)
```

Exemplo de var, no escopo de função (function):

```
// VAR declarado fora da FUNÇÃO, tem escopo GLOBAL
var numero = 30;

// INÍCIO DO CÓDIGO DA FUNÇÃO -----
function exemploVar(){

    //VAR declarado fora da FUNÇÃO, tem escopo de FUNÇÃO
    var mensagem = "Olá mundo!";

    // Funciona, pois "mensagem" está sendo usada no mesmo escopo em que foi declarada
    console.log(mensagem); // Escreva (mensagem)

    // Funciona, pois "numero" tem escopo GLOBAL
    console.log(numero); // Escreva (numero)
}
// FIM DO CÓDIGO DA FUNÇÃO -----

exemploVar(); // Executa o código na função, similar ao que faz o "onclick" do HTML

// ERRO, pois "mensagem" se limita a ser usada na FUNÇÃO onde foi declarada
console.log(mensagem); // Escreva (mensagem)

// FUNCIONA, pois "numero" tem escopo GLOBAL
console.log(numero); // Escreva (numero)
```