Aula 8
Introdução ao JavaScript

Introdução ao JavaScript

JavaScript Interno, Externo, Output, Declarações e Sintaxe

JavaScript é uma linguagem de programação que implementa itens complexos à página, atualizando e modificando as informações, diferente de apenas expô-los na página de forma estática. Junto com HTML e CSS, o JavaScript é uma das três principais tecnologias da web.

1. **HTML é uma linguagem de marcação**, define o conteúdo das páginas web.
2. **CSS é uma linguagem de regras**, define o layout e estilo do conteúdo das páginas.
3. JavaScript, por fim, é uma linguagem de programação, usada para definir os comportamentos dos elementos da página.

Primeiros Passos

JavaScript interno

Quando utilizamos o JavaScript no HTML, o inserimos entre as tags <script> e </script>. Esses scripts podem ser inseridos tanto no body quanto na head do documento HTML.

Vamos primeiro testar como inserir um elemento JavaScript na head. Para isto, prepare o seguinte documento HTML:

```html
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1"
    >
    <title>Usando JS</title>
</head>

<body>

    <p id="p1">Isto é um parágrafo</p>
    <button type="button">Clique aqui</button>

</body>
</html>
```

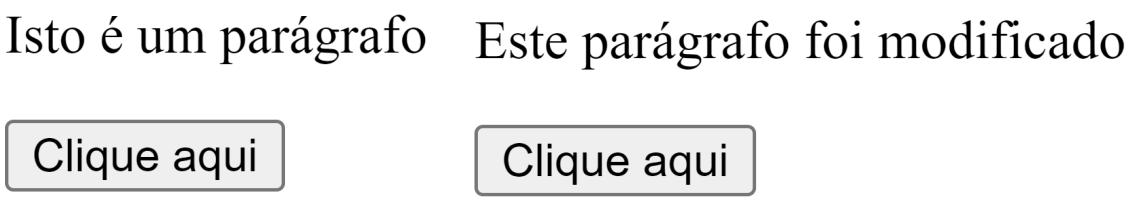
A seguir, vamos adicionar um script à head e dentro dele inserir uma função que será responsável por modificar o parágrafo que escrevemos antes.

```html
<script type="text/javascript">
    function mudarParagrafo()
    {
        document.getElementById("p1").innerHTML = "Este parágrafo foi modificado";
    }
</script>
```

Dentro do botão, precisamos chamar esta função, pois se ela não for chamada, nada acontecerá, por tanto, definimos que ao clicar no botão a função é acionada.

```html
<p id="p1">Isto é um parágrafo</p>
<button type="button" onclick="mudarParagrafo()">Clique aqui</button>
```

Agora, basta testar executando o arquivo na janela do navegador. Clique no botão e verifique se a mudança ocorreu.



Vamos também adicionar um script diretamente no body para exemplificar como funciona.

Recorte as linhas contidas na tag <script>, fazendo com que não aja mais script na head, e cole o código recortado abaixo da tag do botão, da seguinte maneira:

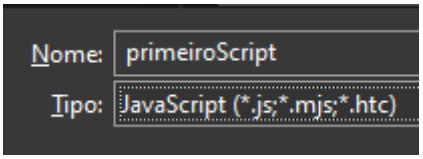
```html
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Usando JS</title>

</head>

<body>

    <p id="p1">Isto é um parágrafo</p>
    <button type="button" onclick="mudarParagrafo()">Clique aqui</button>

    <script type="text/javascript">
        function mudarParagrafo()
        {
            document.getElementById("p1").innerHTML = "Este parágrafo foi modificado";
        }
    </script>

</body>
</html>
```

Ao testar, verá que continua a seguir o mesmo funcionamento. Apesar de ser possível inserir script no arquivo HTML destas duas formas, é muito mais aconselhável utilizar o arquivo JavaScript externo, como veremos na próxima seção.

JavaScript Externo

A utilização do JavaScript de forma externa é mais aconselhável, pois podemos utilizar o mesmo script para vários documentos HTML, sem precisar repetir o mesmo código dentro de cada arquivo HTML.

Primeiro, vamos criar um novo arquivo e salvá-lo como tipo .js, para que seja compatível com JavaScript.



Dentro do arquivo JavaScript, cole a função que utilizamos anteriormente no HTML.

```
index.html  primeiroScript.js
1 function mudarParagrafo()
2 {
3     document.getElementById("p1").innerHTML = "Este parágrafo foi modificado";
4 }
5
```

No arquivo HTML, apague a função que havia antes e deixe apenas a tag Script que fará a referência ao arquivo que acabamos de criar.

```
<body>

    <p id="p1">Isto é um parágrafo</p>
    <button type="button" onclick="mudarParagrafo()">Clique aqui</button>

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

</body>
```

Salve ambos os arquivos e teste novamente no navegador.

Elementos Output

Através do JavaScript, podemos dispor as informações mostradas ao usuário de diversas maneiras. Com isso, vamos aprender quais são elas e de que forma podemos utilizá-las.

Usando InnerHTML

Para testar este exemplo, vamos utilizar JavaScript interno, para facilitar na visualização.

O atributo 'id' é o que define qual elemento HTML está sendo utilizado, já a propriedade 'innerHTML' define qual o conteúdo. Observe que a tag <p> está com seu conteúdo vazio, ela possui apenas um id para identificá-la.

```
<p id="p2"> </p>

<script>
    document.getElementById("p2").innerHTML = 5 + 6;
</script>
```

Ao executar o arquivo no navegador, veremos que no local da tag <p> haverá apenas a equação que foi inserida através do innerHTML.

Usando document.write

Primeiro vamos testá-lo de uma forma que não é utilizado convencionalmente. Escreva o seguinte código e teste em seu navegador:

```
<p>Testando document.write() sozinho</p>

<script>
    document.write(5 + 6);
</script>
```

Agora, vamos utilizá-lo dentro de um botão.

```
<p>Testando document.write() dentro de um botão</p>

<button type="button" onclick="document.write(5 + 6)">Clique Aqui</button>
```

Note que ao testar o arquivo no navegador, após clicar no botão, todo o conteúdo da página será apagado, restando apenas o resultado da equação. Isso ocorre pois utilizamos esse recurso apenas para testes, ele não deve ser acionado após o documento HTML já ter sido carregado, senão ocorrerá como no exemplo, onde o conteúdo da página é apagado.

Usando window.alert()

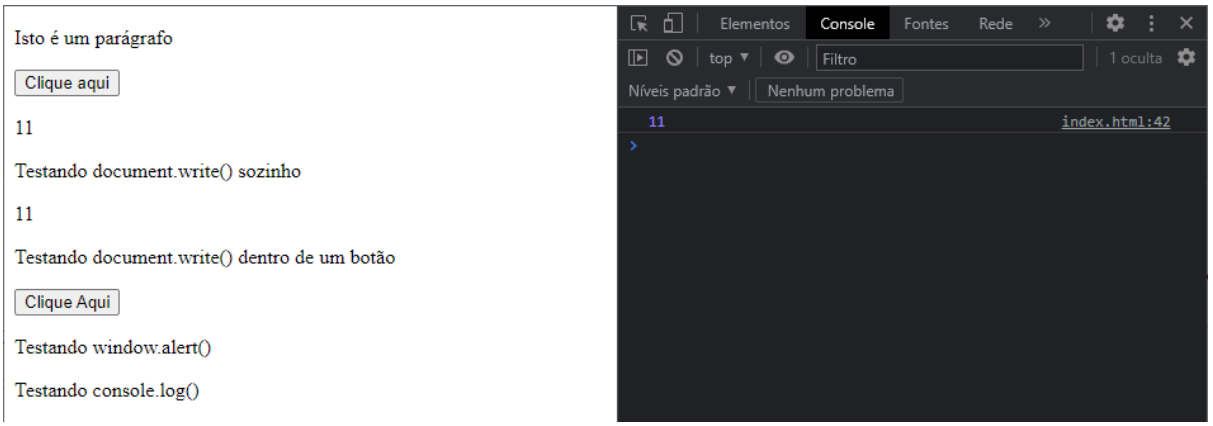
Este outro tipo de Output geralmente vemos quando navegamos na internet, quando sites pedem permissão para ter acesso a algum tipo de dado. É um alerta que abre uma janelinha sobre a aba do navegador.

```
<p>Testando window.alert() </p>
<script>
    window.alert(5 + 6);
</script>
```

Usando console.log()

É necessário entrar no modo de debug para acessar o console. No navegador, pressione F12 no teclado para abrir o painel de debug, selecione a opção 'console' e dê *refresh* na página.

```
<p>Testando console.log() </p>

<script>
    console.log(5 + 6);
</script>
```



Declarações

Uma declaração é uma instrução, um comando dado ao computador pelo programa. Em uma linguagem de programação, essas instruções são chamadas Declarações.

Em JavaScript, temos 5 tipos de declarações, sendo elas:

- valores;
- operadores;
- expressões;
- palavras-chave (keywords);
- e comentários.

Veremos cada uma delas nas aulas seguintes.

Ponto e Vírgula

Quando escrevemos declarações, precisamos sinalizar onde cada uma termina, para isso utilizamos ponto e vírgula ';' ao final de cada uma.

```
<script>
    document.write(1 + 2);
    document.write(3 + 4);
    document.write(5 + 6);
</script>
```

Também podemos colocar todas as declarações em um linha só e separá-las com ponto e vírgula. Nesse caso, ao executar no navegador os três resultados ficarão todos juntos, geralmente usa-se declarações na mesma linha para criação de variáveis, pois o valor delas não aparecerá amontoado na mesma linha.

```
<script>
    document.write(1 + 2); document.write(3 + 4); document.write(5 + 6);
</script>
```

Espaços em Branco

Linguagens de Programação no geral, ignoram excesso de espaço em branco, o mesmo ocorre com JavaScript. Então utilizamos os espaços em branco no código para organização e otimização da leitura, como em declarações com aspas e uso de operadores.

```
<p>Testando espaço em branco</p>
<script>
    document.write("Sem Espaço");
</script>

<p>Testando espaço em branco</p>
<script>
    document.write( "Com Espaço" );
</script>
```

Blocos de código

As declarações em JavaScript são organizadas em blocos de código, deste modo sabe-se quais declarações devem ser executadas juntas em uma função, por exemplo. Os blocos iniciam e terminam utilizando chaves { }.

Há duas maneiras de posicionar as chaves: abaixo do nome da função, de maneira que as duas chaves fiquem alinhadas verticalmente; ou após o nome da função, na mesma linha. As duas maneiras estão corretas e dependem apenas de como o programador prefere organizar o código.

```
function mudarParagrafo()
{
    document.getElementById("p1").innerHTML = "Este parágrafo foi modificado";
}


function mudarParagrafo(){
    document.getElementById("p1").innerHTML = "Este parágrafo foi modificado";
}
```

Sintaxe

A sintaxe de JavaScript define 2 tipos de valores: literais, que possuem valores fixos que não são modificados; e variáveis, que podem ser modificados, acrescentados ou subtraídos.

Literais

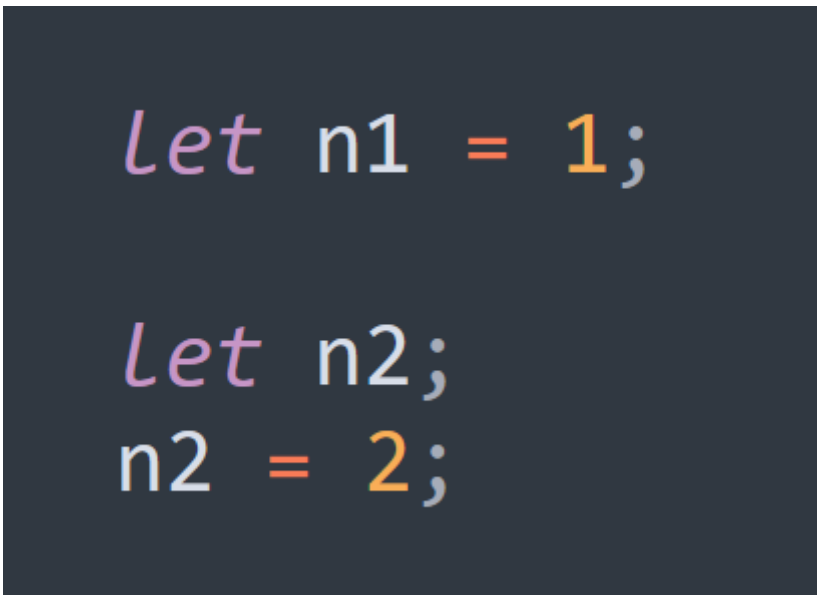
Para os valores literais temos números e textos. Os números podem ser tanto números inteiros quanto números decimais. Os textos por sua vez, precisam ser sinalizados como textos, para isso utilizamos aspas, podem ser simples ou duplas.

```
<script>
    document.write(100);
    document.write(10.1);
    document.write("Eu sou uma String");
    document.write('Eu também');
</script>
```

Variáveis

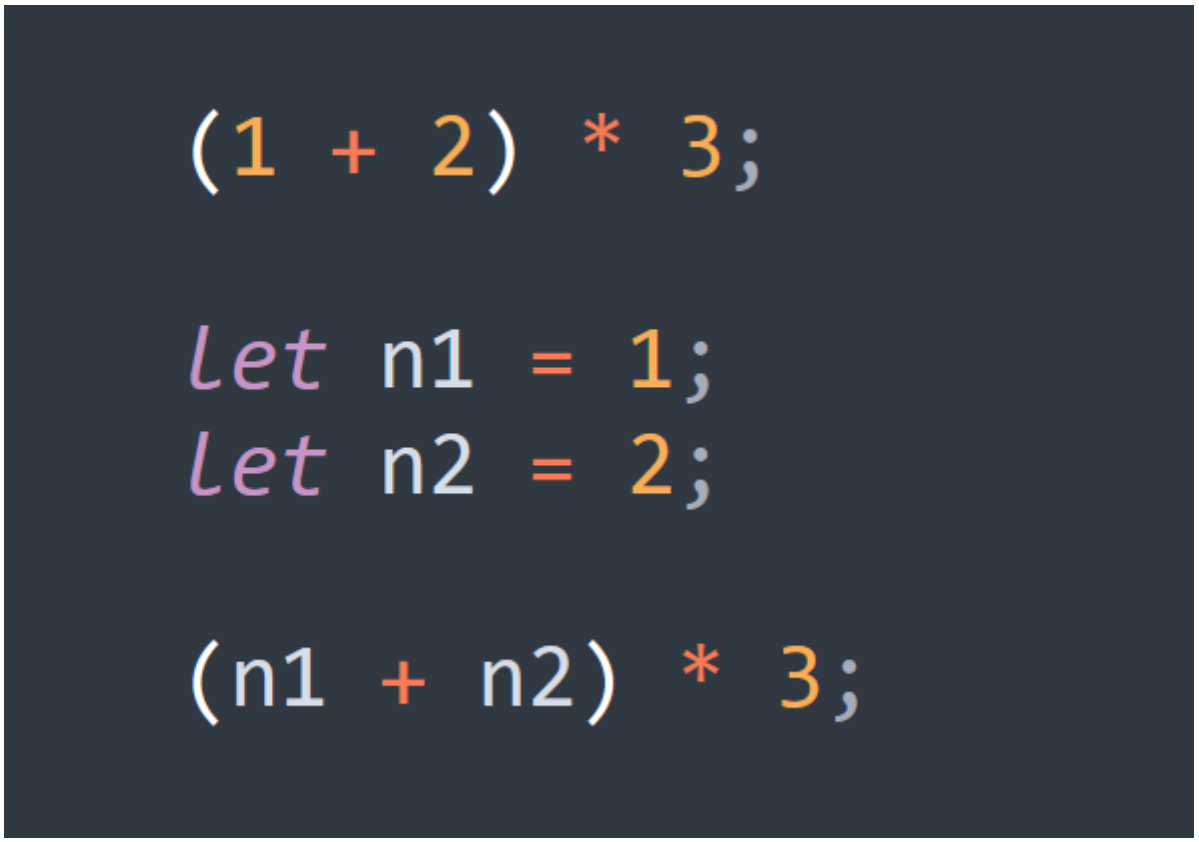
Em uma linguagem de programação, variáveis são utilizadas para guardar valores. Em JavaScript usamos ‘var’, ‘let’ e ‘const’ como palavras-chave para declarar variáveis. Após o nome da variável, é utilizado o sinal de igual ‘=’ para fazer a atribuição do valor.

Também é possível declarar a variável primeiro e atribuir o valor na declaração seguinte, as duas formas estão corretas.



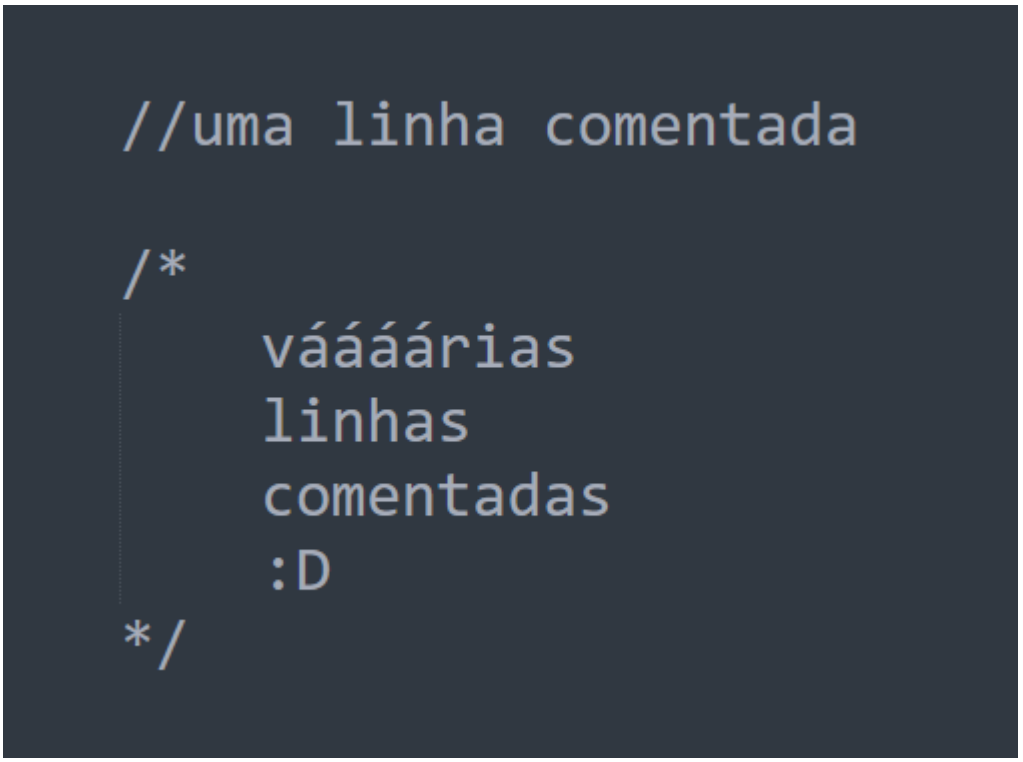
Operadores

Operadores aritméticos são utilizados para realizar as equações e expressões para computar valores. Podem ser utilizados tanto com os números quanto com variáveis numéricas.



Comentários

Para fazer anotações no código, usamos os comentários. Podemos comentar apenas uma linha ou múltiplas linhas:



Nomes e identificadores

Identificadores são utilizados para nomear variáveis e funções. Para que funcionem corretamente devem seguir algumas regras:

- Iniciar com letras A-Z ou a-z;
- iniciar com cifrão $;
- iniciar com underline
- não podem iniciar com numerais;
- não podem conter hífen ou símbolos de operações aritméticas;
- não pode conter pontuações e acentuações.

Case sensitive

Case Sensitive significa que há diferença entre letras minúsculas e maiúsculas. Este é um ponto importante para a criação de nomes de variáveis e identificadores:

- uma variável chamada ‘numdois’
- é diferente de uma chamada ‘numDois’.

Camel Case

Quando vamos utilizar um nome para uma variável ou identificador é importante que possamos ler seu nome e entendê-lo, pois ele nos dará uma breve descrição de para o que estamos a utilizando. Esses nomes não podem conter espaço, com isso, os programadores têm utilizado diversas formas de agrupar palavras juntas, sabendo onde termina uma e começa a outra:

- Usando underline:
  - nome_de_variavel_1
  - nome_cidade
  - nome_pessoa

- Upper Camel Case
  - NomeDeVariavel1
  - NomeCidade
  - NomePessoa

- Lower Camel Case
  - nomeDeVariavel1
  - nomeCidade
  - nomePessoa

Tarefa de Casa

Assinale os nomes de variáveis INCORRETAS e justifique o por quê.

⬜ nomeSobrenome
⬜ _contador
⬜ alfa_beta_rotina
⬜ número2
⬜ variavel
⬜ x
⬜ %dividir
⬜ 1numero
⬜ _texto_
⬜ 6_05 _1312
⬜ _var1
⬜ Reinicializa
⬜ Xs
⬜ A$
⬜ teclado_lidinho
⬜ Germa66
⬜ 1°_lugar
⬜ PlayerID
⬜ tempo e hora
⬜ km/h
⬜ raio.circulo
⬜ cont
⬜ d’agua
⬜ primeiro
⬜ TesteUm
⬜ 2dias
⬜ $dolar
⬜ teste 1