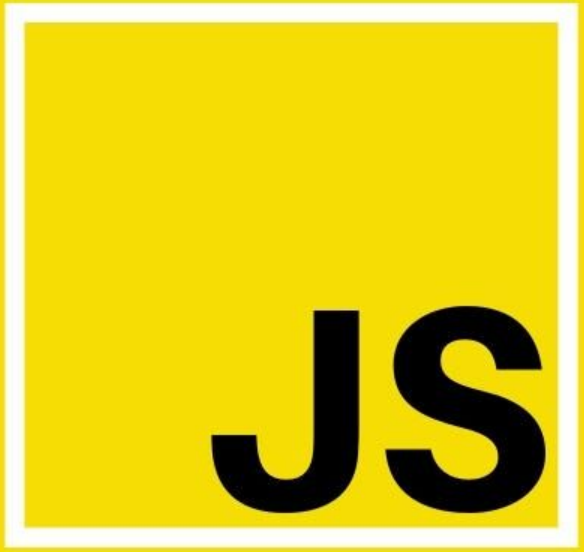


JAVASCRIPT

```
49      <form name="form">
50      <input type=text name="texto" class="caixadetexto">
51      <input type=button onClick="verificarTexto()" class="botao" value="Clique Aqui">
52      </form>
53      <script>
54        function verificarTexto(){
55    if (form.texto.value == "")
56    {alert ("Não há texto escrito") }
57    }
58    </script>
```

```
<form name="form">
  <input type=text name="texto" class="caixadetexto">
  <input type=button onClick="verificarTexto()" class="botao"
value="Clique Aqui">
  </form>
  <script>
    function verificarTexto(){
if (form.texto.value == " ")
{alert ("Não há texto escrito") }
}
</script>
```

Para o Front End

Sintaxe

Variáveis var e let e const

Funções

Estruturas de decisão (condicionais if, else, else if e switch);

Estruturas de repetição (For, while, do-while);

Funções assíncronas (Async, Await)

Escopos locais e globais

DOM (Document Object Model)

Como inserir no HTML

<head>

<script type="text/javascript" src="arquivo.js">

</script>

</head> **ou**

<body>

<script>

</script>

</body>

Tipos de dados

- **Boolean** – possuem apenas dois valores: verdadeiro ou falso;
- **Undefined** – indica que não foi definido um valor;
- **Null** – indica que um valor é nulo;
- **Number** – armazena valores numéricos;
- **String** – armazena textos;
- **Symbol** – armazena símbolos;

Sintaxe básica

JavaScript é case-sensitive e usa o conjunto de caracteres Unicode. Por exemplo, a palavra ‘Chuva’ pode ser usada como nome de variável.

var Chuva = "chovendo";

Ou seja, se essa variável for declarada posteriormente como ‘chuva’ não irá funcionar.

Por exemplo, considerando o seguinte código:

JavaScript

```
1 let name = "John";
2
3 if (name == "john") {
4   console.log("Hello, John!");
5 } else {
6   console.log("I don't know you.");
7 }
8
```

Verifique!

```
<script>
let name = "John";
if (name == "john") {
  console.log("Hello, John!");
} else {
  console.log("I don't know you.");
}
</script>
```

Neste caso, a variável name é definida como “John” com uma letra maiúscula. No entanto, na instrução if, estamos comparando a variável com a string “john”, com uma letra minúscula. Como a linguagem é sensível a maiúsculas e minúsculas, essa comparação resultará em false e a mensagem “I don’t know you.” será exibida no console.

Sintaxe básica

O exemplo a seguir mostra a aparência de uma instrução JavaScript:

JavaScript

```javascript
var x = 5;
var y = 10;
var sum = x + y;
document.write(sum); // Imprime o valor da variável
```

Verifique!

```
<script>

var x = 5; // Variável no valor de 5

var y = 10; // Variável no valor de 10

var sum = x + y; // Variável que soma as duas anteriores

document.write(sum); // Imprime na tela o valor da variável

</script>
```

Comentários JavaScript

Os comentários em JavaScript são linhas de código que são ignoradas pelo interpretador da linguagem e servem para adicionar explicações ou observações ao código. Então, eles podem ser úteis para deixar o código mais legível e fácil de entender, especialmente quando trabalhamos em projetos mais complexos ou em equipe.

Existem dois tipos de comentários em JavaScript:

Comentários de linha única: são utilizados para adicionar um comentário em uma única linha de código. Eles são representados por um símbolo de barra dupla (//) seguido pelo texto do comentário.

Por exemplo:

```
// Este é um comentário de linha única

let x = 10; // Este é outro comentário de linha única
```

Comentários de múltiplas linhas: são utilizados para adicionar um comentário que ocupa várias linhas de código. Então, eles são representados por um símbolo de barra e asterisco de abertura (/) seguido pelo texto do comentário e um símbolo de asterisco e barra de fechamento (/) no final.

Por exemplo:

/*

Este é um comentário de múltiplas linhas.

Ele pode ser útil para adicionar explicações mais detalhadas ou observações ao código.

****/***

let y = 20;

Declarações

Existem três tipos de declarações em JavaScript.

var - Declara uma variável, opcionalmente, inicializando-a com um valor.
Experimental
let - Declara uma variável local de escopo do bloco, opcionalmente, inicializando-a com um valor.
Experimental
const - Declara uma constante de escopo de bloco, normalmente apenas de leitura.

Declarando variáveis

Você pode declarar uma variável de quatro formas:

- Com a palavra chave **`var`**. Por exemplo, `var x = 42`. Esta sintaxe pode ser usada para declarar tanto variáveis locais como variáveis globais.
- Por simples adição de valor. Por exemplo, **`x = 42`**. Isso declara uma variável global. Essa declaração pode gerar um aviso de advertência no JavaScript. Essa variante não é recomendada.
- Com a palavra chave **`let`**. Por exemplo, `let y = 13`. Essa sintaxe pode ser usada para declarar uma variável local de escopo de bloco.
- Usando **const** para declarações fixas, mas que podem ser mutáveis.

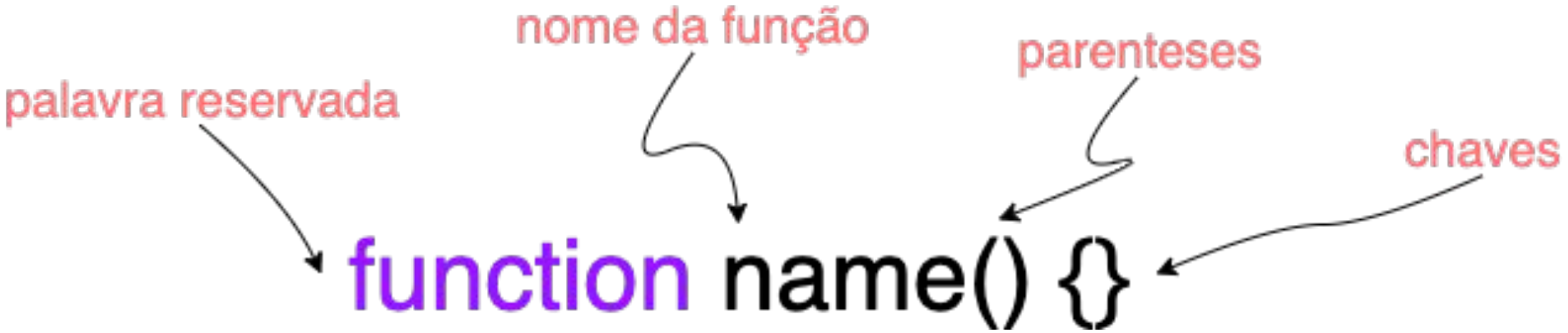
Funções

Funções são blocos de construção fundamentais em JavaScript. Uma função é um procedimento de JavaScript - um conjunto de instruções que executa uma tarefa ou calcula um valor. Para usar uma função, você deve defini-la em algum lugar no escopo do qual você quiser chamá-la. Ou seja, ela é executada quando for solicitado!

```
// declaração
function saudacao(){
    console.log("Olá!");
}

// chamada
saudacao()
```

```
Olá!
```


palavra reservada
nome da função
parenteses
chaves
function name() {}

Função do tipo Void (vazia);

```
<script>
function mostrarAlgo () {
console.log ('estou na função'); //mostra resultado no console
document.write ('estou na função') //mostra resultado na tela
}
mostrarAlgo (); //exibe o resultado porque foi convocada
</script>
```

Função com parâmetro

```
<script>
        function soma (numero1, numero2) {
        var somaDosNumeros = numero1 + numero2;
        console.log(somaDosNumeros);
        }
        soma(12,5);
        soma(6,8)
</script>
```

Em JavaScript, os parâmetros são valores que você pode passar para uma função quando a chama. Os parâmetros são como "entradas" que uma função precisa para executar uma tarefa específica. Imagine uma função como uma máquina que realiza uma tarefa, e os parâmetros são os dados de entrada que você coloca nessa máquina para que ela possa fazer algo útil.

Função com return

```
<script>

function soma (numero1, numero2) {

var somaDosNumeros = numero1 + numero2;

return somaDosNumeros;

}

let numerosAparecer = soma (12,3);

console.log (numerosAparecer);

document.write (numerosAparecer);

</script>
```

A função return é uma parte fundamental do JavaScript que permite que uma função retorne um valor específico após a execução. Isso é útil quando você deseja que uma função produza um resultado que pode ser usado em outras partes do seu código.

- Você pode ter múltiplas instruções return em uma função, mas a função retornará apenas o primeiro que for executado.
- Uma vez que a função atinge uma instrução return, ela sai imediatamente da função, e qualquer código após o return não será executado.
- O valor retornado pode ser de qualquer tipo de dados, como números, strings, objetos, arrays, etc.

Estruturas de decisão

Estruturas de decisão são muito utilizadas na programação. Sua função é fazer com que sistemas tomem decisões a partir de uma expressão.

No JavaScript, temos as seguintes estruturas condicionais:

- **IF/Else:** especificando um bloco de código para ser executado de acordo com a condição.
- **Else if**: especificar uma nova condição de teste caso a primeira seja falsa.
- **Switch**: especificar diversos blocos alternativos para serem executados.

Decisão simples

```
<script>
        let name = "John";
        if (name == "john") {
          console.log("Hello, John!");
        } else {
          console.log("I don't know you.");
          alert("I don't know you.")
        }
 </script>
```

Decisão composta

```
<script>
        let sinalSemaforo = "verde";
        if (sinalSemaforo == "verde") {
          console.log("Pode avançar");
          document.write ("Pode avançar");
        } else if (sinalSemaforo == "amarelo") {
          console.log ("Piá, acelere que vai ficar vermelho!");
          document.write ("Piá, acelere que vai ficar vermelho!");
        } else {
          console.log ("Já era, ficou vermelho!");
          document.write ("Já era, ficou vermelho!");
        }
        </script>
```

Switch

switch (expressao) {

case valor1: // Código a ser executado se a expressão for igual a valor1

break; case

valor2: // Código a ser executado se a expressão for igual a valor2

break;

// Mais casos...

default: // Código a ser executado se a expressão não coincidir com nenhum dos casos anteriores

}

O switch é uma estrutura de controle de fluxo em JavaScript que permite que você avalie uma expressão em relação a vários casos e execute diferentes blocos de código com base no valor da expressão. É útil quando você deseja tomar decisões com base em múltiplos valores possíveis.


To Be Continued...