



# String

## **R**esumo

Uma das características mais fundamentais de uma linguagem de programação é o conjunto de tipos de dados que ela suporta. Esses são os tipos de valores que podem ser representados e manipulados em uma linguagem de programação.

JavaScript permite que você trabalhe com três tipos de **dados primitivos**:

- **Números**, exemplo. 123, 120,50 etc.
- **Strings** de texto, exemplo “Esta string de texto” etc.
- **Booleanos**, exemplo verdadeiro ou falso.

JavaScript também define dois tipos de dados triviais, **null** (nulo) e **undefined** (indefinido), cada um dos quais define apenas um único valor. Além desses tipos de dados primitivos, o JavaScript oferece suporte a um tipo de dados composto conhecido como **objeto**.

**Observação** - JavaScript não faz distinção entre valores inteiros e valores de ponto flutuante. Todos os números em JavaScript são representados como valores de ponto flutuante. JavaScript representa números usando o formato de ponto flutuante de 64 bits definido pelo padrão IEEE 754.

## Variáveis em JavaScript



Como muitas outras linguagens de programação, JavaScript tem variáveis. As variáveis podem ser consideradas contêineres nomeados. Você pode colocar dados nesses contêineres e, em seguida, referir-se aos dados simplesmente nomeando o contêiner.

Antes de usar uma variável em um programa JavaScript, você **pode declará-la**. As variáveis são declaradas com a palavra-chave **var** da seguinte maneira:

```
var nome;
```

```
var sobrenome = "Silva"
```

Você também pode declarar várias variáveis com a mesma palavra-chave **var** da seguinte maneira:

```
Var nome, sobrenome;
```

O armazenamento de um valor em uma variável é chamado de **inicialização** de variável. Você pode fazer a inicialização da variável no momento da criação da variável ou posteriormente, quando precisar dessa variável.

Por exemplo, você pode criar uma variável chamada "salario" e atribuir o valor 7.000,50 a ela mais tarde. Para outra variável, você pode atribuir um valor no momento da inicialização da seguinte maneira:

```
Var salario;
```

```
Var Nome = "Luis"
```

Salario = 3000



**Nota** - Use a palavra-chave **var** apenas para declaração ou inicialização, uma vez durante a vida de qualquer nome de variável em um documento. Você não deve declarar novamente a mesma variável duas vezes.

JavaScript é uma linguagem **não tipada**. Isso significa que uma variável JavaScript pode conter um valor de qualquer tipo de dados. Ao contrário de muitas outras linguagens, você não precisa dizer ao JavaScript durante a declaração da variável que tipo de valor a variável manterá. O tipo de valor de uma variável pode mudar durante a execução de um programa e o JavaScript cuida disso automaticamente.

## Nomes de variáveis JavaScript

Ao nomear suas variáveis em JavaScript, mantenha as seguintes regras em mente.

- Não usar nenhuma das palavras-chave reservadas de JavaScript como um nome de variável. Por exemplo, os nomes das variáveis `break` ou `boolean` não são válidos.
- Os nomes das variáveis JavaScript não devem começar com um numeral (0-9). Eles devem começar com uma letra ou um caractere de sublinhado. Por exemplo, `123test` é um nome de variável inválido, mas `_123test` é válido.
- Os nomes das variáveis JavaScript diferenciam maiúsculas de minúsculas. Por exemplo, `Nome` e `nome` são duas variáveis diferentes.

## Conteúdo Bônus



O JavaScript é bastante poderoso no manuseio de strings (textos, caracteres), fornecendo ao programador total flexibilidade, disponibilizando funções que podem auxiliar no desenvolvimento.

Uma das opções é usar o `length` para retornar o tamanho da string (números de caracteres)

Exemplo:

```
var curso = "Javascript"
```

```
console.log(curso.length) //retorna 10
```

Outra opção é usar o `charAt` que retorna o caractere da posição especificada (inicia em 0).

```
var curso = "Javascript"
```

```
console.log(curso.charAt(2) //retorna v
```

Um dos mais usados é o `indexOf`, que retorna o número da posição onde começa a primeira "string".

```
var curso = "Javascript top"
```

```
curso.indexOf("top")); //Resultado: 11
```

Outro que podemos usar é a `substring` que retorna o conteúdo da string que corresponde ao intervalo especificado. Começa no caractere posicionado em `index1` e termina em `index2`  
`substring(index1, index2)`

```
var curso = "Javascript"
```



```
curso.substring(0,4); //Resultado: Java
```

Outro muito legal é o `TOUPPERCASE()` que passa o conteúdo da string para letra maiúscula (Caixa Alta)

```
var curso = "Javascript"
```

```
curso.toUpperCase(); //Resultado: JAVASCRIPT
```

Outro muito legal também é o `TOLOWERCASE()` que passa o conteúdo da string para letra minúscula (Caixa Baixa).

```
var curso = "Javascript"
```

```
curso.toLowerCase() //Resultado: javascript
```

Outro que usamos bastante no nosso dia a dia de desenvolvedor é o `replace`. Ele substitui um valor por outro

```
var curso = "Javascript"
```

```
curso.replace("JavaScript","Java");//Resultado: Java
```

## Referência Bibliográfica

FLANAGAN, David. **JavaScript: O Guia Definitivo**. 6ª Ed. Porto Alegre: Bookman, 2013.

FREEMAN, Eric. **Use a cabeça!: programação JavaScript**. 1ª Ed. São Paulo: Alta Books, 2016

## ATIVIDADE PRÁTICA



**Título da Prática:** Como trabalhar com String?

**Objetivos:** Compreender diversos métodos de string

**Materiais, Métodos e Ferramentas:** Para realizar esta prática vamos utilizar o Visual Studio Code

### Prática

Olá, aluno! Neste exercício, vamos trabalhar com um código em JavaScript que lida com uma lista de nomes. Nosso objetivo é aplicar uma transformação específica nos nomes, convertendo-os para letras maiúsculas, mas apenas se tiverem um tamanho menor que 5 caracteres. Ou seja, nomes curtos serão modificados, enquanto nomes mais longos permanecerão inalterados.

Para resolver esse problema, podemos utilizar alguns conceitos fundamentais da linguagem JavaScript, como arrays, iteração e manipulação de strings. Vamos usar o método `map()` para iterar sobre cada elemento da lista de nomes e, em seguida, vamos verificar o tamanho de cada nome com a propriedade `length`. Se o nome tiver menos de 5 caracteres, aplicaremos o método `toUpperCase()` para convertê-lo para letras maiúsculas. Caso contrário, manteremos o nome original.

É importante lembrar que, para implementar essa solução, é necessário ter um conhecimento básico de JavaScript e de como trabalhar com arrays e strings. Agora, vamos colocar em prática esses conceitos e resolver o problema de converter nomes para letras maiúsculas com base no tamanho do nome. Mãos à obra!

### Resolução:

```
```\njavascript\n// Array com os nomes das pessoas\nlet nomes = ["joão", "maria", "carlos", "ana", "pedro"];\n\n// Iterar sobre cada nome e convertê-lo para letras maiúsculas\nlet nomesMaiusculos = nomes.map(nome => nome.toUpperCase());\n\n// Exibir os nomes em letras maiúsculas\nconsole.log(nomesMaiusculos);\n```\n
```



Neste exemplo, o método `toUpperCase()` é aplicado a cada elemento do array `nomes` usando o método `map()`. O resultado é armazenado no array `nomesMaiusculos`. Em seguida, os nomes convertidos em letras maiúsculas são exibidos no console.

A saída será:

```
```\n["JOÃO", "MARIA", "CARLOS", "ANA", "PEDRO"]\n```\n
```

Dessa forma, você pode aplicar essa lógica ao seu código JavaScript para converter os nomes para letras maiúsculas.

**Ir para exercício**