String

Depois que a sequência de caracteres for definida, a string é imutável, ou seja, não poderá ter seu valor alterado.

Sempre que manipulamos uma string, é criada uma nova instância dela por baixo dos panos, o que significa que será gerado um novo espaço na memória com uma cópia do valor da string. Por isso, temos que utilizar uma variável para armazená-la.

No js existe o recurso o objeto global *string* que permite criar ou converter um tipi em uma string.

É possível interpolar, concatenar, checar posições de caracteres ou ainda substituir partes de strings.

- Concatenar: juntar duas strings (somar)

- Interpolar: template strings, templates literais (`${}`)

**Métodos para strings**

O JS diferencia strings como tipos primitivos de objetos Strings, mas no final, mesmo as strings criadas da forma literal (*const texto = ‘texto’*), acaba convertida para um objeto do tipo string. Por isso, temos acesso a uma série de métodos e propriedades deste objeto.

*.length ->* informa o tamanho

*.charAt()->* acessa a posição de um caractere de uma string

*.indexOf()->* retorna a posição de um caractere dentro da string. Retorna somente a primeira ocorrência.

*.toUpperCase()/.toLowerCase()->* deixa maiúsculas ou minúsculas

*.substr()->* substring, permite que façamos a extração de parte de uma string. Recebe como parâmetro o início e o fim da nova string

*.slice()->* retornará parta de uma string, desde que passemos nos parâmetros o índice de início e de fim.

*.replace()->* substitui parte de uma string por outra. Recebe como parâmetro a string a ser substituída e a que será colocada no lugar

*.concat()->* concatena strings sem usar o operador ‘+’. Adiciona a nova string ao fim da anterior

*.split()->* quebra uma string com base em caracteres separadores informados como parâmetro. Gera como resultado um array de strings

*.trim()->* remove os espaços em branco no início ou fim de uma string. Variações: *.trimEnd(); trimStart(); trimLeft(); trimRight()*