11 - Strings

Características das Strings

Strings são mais do que apenas textos. No JavaScript, elas são objetos que vêm com um monte de métodos úteis.

Imutabilidade das Strings

```
let saudacao = "Olá";
saudacao[0] = 'A';
console.log(saudacao); // Olá
```

As strings são imutáveis. Isso significa que, uma vez criada, você não pode alterar um caractere individualmente. Tentar fazer isso não vai gerar um erro, mas também não vai mudar a string.

Concatenação de Strings

```
let nome = "Maria";
let mensagem = "Oi, " + nome + "!";
console.log(mensagem); // Oi, Maria!
```

Aqui estamos juntando, ou concatenando, strings usando o operador `+`. É uma forma básica de construir strings a partir de outras strings.

Métodos Avançados de String

JavaScript oferece vários métodos para trabalhar com strings de maneira mais sofisticada.

Método `slice()`

```
let frase = "Aprendendo JavaScript em 2023!";
let ano = frase.slice(-5);
console.log(ano); // 2023!
```

`slice()` **corta e retorna uma parte da string. Aqui,** `-5` **significa que estamos começando do quinto caractere a contar do fim.**

Método `replace()`

```
let novaFrase = frase.replace("2023", "2024");
console.log(novaFrase); // Aprendendo JavaScript em 2024!
```

`replace()` **substitui um trecho da string por outro. Neste exemplo, substituímos "2023" por "2024".**

Método `toUpperCase()` e `toLowerCase()`

```
let grito = "olá".toUpperCase();
console.log(grito); // OLÁ

let sussurro = "OI".toLowerCase();
console.log(sussurro); // oi
```

Estes métodos transformam a string em maiúsculas ou minúsculas, respectivamente.

Método `trim()`

```
let textoComEspacos = "   Olá, mundo!   ";
let textoSemEspacos = textoComEspacos.trim();
console.log(textoSemEspacos); // Olá, mundo!
```

`trim()` **remove espaços em branco do início e do fim da string.**

Strings e Caracteres Especiais

Strings podem conter caracteres especiais, que são úteis para várias situações.

Quebras de Linha e Tabulações

```
let poema = "Roses are red,\nViolets are blue.";
console.log(poema);
// Roses are red,
// Violets are blue.
```

`\n` **cria uma nova linha. É útil para formatar textos longos ou poesia.**

Caracteres de Escape

```
let citação = "Ela disse: \"JavaScript é incrível!\"";
console.log(citação); // Ela disse: "JavaScript é incrível!"
```

Usamos a barra invertida `\` para incluir aspas dentro de uma string sem encerrar a string.

Template Literals

Template literals são uma maneira mais poderosa e flexível de trabalhar com strings.

```
let produto = "Node.js";
let versao = "v14.17.0";
let descricao = `Estudando ${produto} na versão ${versao}`;
console.log(descricao); // Estudando Node.js na versão v14.17.0
```

Aqui, usamos acentos graves (`) e `${}` para inserir variáveis e expressões dentro da string. Isso torna a concatenação mais fácil e legível.

Exercícios para você ficar fera:

1. **Criando um Convite: Use a concatenação de strings para criar um convite. Inclua o nome do destinatário, o tipo de evento e a data, usando variáveis.**
2. **Diário de Bordo: Escreva um pequeno diário de bordo usando template literals, incluindo data, local e uma descrição do que aconteceu no dia.**
3. **Redator de Notícias: Crie uma string que represente uma notícia, utilizando o método `replace()` para substituir um fato errado por um correto.**
4. **Caixa de Comentários: Use o método `trim()` para limpar os comentários dos usuários antes de publicá-los em um blog ou fórum.**
5. **Carta para um Amigo: Utilize `\\n` para formatar uma carta para um amigo, com saudação, corpo da mensagem e despedida, cada um em uma nova linha.**
6. **Citações Famosas: Crie uma string que inclua uma citação famosa de uma pessoa, utilizando caracteres de escape para incluir aspas na citação.**