Propiedades y métodos de strings

DigitalHouse>

Certified Tech Developer
The Ultimate Degree

Para JavaScript los strings son como una **colección** de caracteres.
Por esta razón disponemos de **propiedades** y **métodos** muy útiles a la hora de trabajar con la información que hay adentro.

Los strings en JavaScript

En muchos sentidos, para JavaScript, un **string** no es más que un **array de caracteres**. Al igual que en los arrays, la primera posición siempre será 0.

```
{}  let nombre = 'Fran';
```

Para acceder a un carácter puntual de un string, nombramos al string y, **dentro de los corchetes**, escribimos el **índice** al cual queremos acceder.

```
{}  nombre[2];
    // accedemos a la letra a, el índice 2 del string
```

.length

Esta **propiedad** retorna la **cantidad total de caracteres** del string, incluidos los espacios.
Al ser una propiedad (veremos más sobre ellas en clases siguientes), solo debemos llamarla, sin necesidad de los paréntesis.

```
{}
let miSerie = 'Mad Men';
miSerie.length; // devuelve 7

let arrayNombres = ['Bart', 'Lisa', 'Moe'];
arrayNombres.length; // devuelve 3

arrayNombres[0].length; // Corresponde a 'Bart', devuelve 4
```

.indexOf()

Busca, en el string, el string que recibe como parámetro.

- **Recibe** un elemento a buscar en el array.
- **Retorna** el primer índice donde encontró lo que buscábamos. Si no lo encuentra, retorna un -1.

```
{}
let saludo = '¡Hola! Estamos programando';

saludo.indexOf('Estamos'); // devuelve 7
saludo.indexOf('vamos'); // devuelve -1, no lo encontró
saludo.indexOf('o'); // encuentra la letra 'o' que está en la posición 2, devuelve 2 y corta la ejecución
```

A diferencia de las propiedades, llamamos **métodos** a las funciones que se encuentran dentro de **objetos** (los veremos en detalle en las próximas clases). A estos **métodos debemos invocarlos** como lo haríamos al llamar una función, con sus **paréntesis y parámetros** (si fuese necesario).

.slice()

Corta el string y devuelve una parte del string donde se aplica.

- **Recibe 2** números como parámetros (pueden ser negativos):
  - El índice desde donde inicia el corte.
  - El índice hasta donde hacer el corte (es opcional).
- **Retorna** la parte correspondiente al corte.

```
{}
let frase = 'Breaking Bad Rules!';

frase.slice(9,12); // devuelve 'Bad'
frase.slice(13); // devuelve 'Rules!'
frase.slice(-10); // ¿Qué devuelve? ¡A investigar!
```

.trim()

Elimina los espacios que estén al principio y al final de un string.

- **No recibe** parámetros.
- No quita los espacios del medio.

```
{}
let nombreCompleto = '   Homero Simpson   ';
nombreCompleto.trim(); // devuelve 'Homero Simpson'

let nombreCompleto = '   Homero   J.   Simpson   ';
nombreCompleto.trim(); // devuelve 'Homero   J.   Simpson'
```

.replace()

Reemplaza una parte del string por otra.

- **Recibe** dos strings como parámetros:
  - El string que queremos buscar.
  - El string que usaremos de reemplazo.
- **Retorna** un nuevo string con el reemplazo.

```
{}
let frase = 'Aguante Python!';
frase.replace('Python', 'JS'); // devuelve 'Aguante JS!'
frase.replace('Py', 'JS'); // devuelve 'Aguante JSthon!'
```

.split()

Divide un string en partes.

- **Recibe** un string que usará como separador de las partes.
- **Devuelve** un array con las partes del string.

```
{}
let cancion = 'And bingo was his name, oh!';

cancion.split(' ');
// devuelve ['And', 'bingo', 'was', 'his', 'name,' , 'oh!']

cancion.split(', ');
// devuelve ['And bingo was his name', 'oh!']
```

Si bien **cada método** realiza una **acción muy simple**, cuando los combinamos podemos **lograr resultados** mucho más **complejos y útiles**.

DigitalHouse>