DigitalHouse >
Coding School

MÉTODOS DE
STRINGS

“

Javascript nos ofrece varios **métodos** y **propiedades** para trabajar con los **strings**.

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.

```
var nombre = 'Fran';
             0123
```

Así, podemos acceder al contenido de la variable nombre utilizando la misma sintaxis que con arrays: *variable[indice]*

```
nombre[3]; // devuelve 'n'
```

.length

Tal vez la conozcan de **arrays**. Esta propiedad retorna la **cantidad total** de los caracteres del string, incluídos los espacios.

Como es una propiedad, al invocarla no necesitamos los paréntesis.

```
var miSerie = 'Mad Men';
miSerie.length; // devuelve 7

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

¡SPOILER ALERT!

Antes de conocer los **métodos de los arrays**, es necesario espiar un poco la definición de **función** y de **método**, que veremos a fondo más adelante.

Una **función** es un bloque de código que nos permite ejecutar una tarea tantas veces como necesitemos. Las funciones pueden **recibir datos** para realizar la tarea y estos se conocen como **parámetros**.

Cuando una función le pertenece a un objeto, en este caso nuestro array, la llamamos **método**.

.indexOf()

Este método busca en el string donde se aplica, el string que recibe como parámetro.

En el caso de **encontrarlo**, retorna la primera **posición** donde lo encontró.

En el caso de **no encontrarlo**, retorna un **-1**.

```
var 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
```

.slice()

Este método **corta y devuelve una parte del string** donde se aplica.

Recibe 2 números como parámetros:

- El índice **desde donde** inicia el corte.
- El índice **hasta donde** hacer el corte y **es opcional**.

Ambos índices pueden recibir números negativos.

```
var frase = 'Breaking Bad Rules!';

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

.trim()

Este método **elimina los espacios** que estén al principio y al final de un string.

Si hay espacios en el medio, no los quita.

No recibe parámetros.

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

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

.split()

Este método **divide un string** en varios strings utilizando el string que le pasemos como separador.

Devuelve un array con las partes del string original.

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

.replace()

Este método **reemplaza una parte de un string** por otra.

Recibe dos parámetros:

- El string que queremos buscar
- El string que usaremos de reemplazo

Devuelve un **string nuevo** con esa modificación.

```
var frase = 'Aguante Phyton!'
frase.replace('Phyton','JS') //devuelve 'Aguante JS!'
frase.replace('Phy','JS') //devuelve 'Aguante JSton!'
```

“

Si bien **cada método** realiza una **acción muy simple**.

Cuando los **combinamos**, podemos lograr **resultados** mucho **más complejos** y **útiles**.