Strings JS

Los datos primitivos de tipo **string** se utilizan para almacenar datos en forma de texto. Los datos de tipo strings se especifican usando comillas simples, dobles o comillas invertidas.

```
const name = "Peter";
const surname = 'Parker';
const location = `New York`;
const quote = "He is 'Spider-Man'";
```

Como podéis ver en el último ejemplo cabe la posibilidad de combinar **comillas dobles** con **comillas simples** y viceversa a la hora de insertar dichas comillas como valor, siempre y cuando usemos el mismo tipo de comillas para englobar todo el valor y no repetir su uso en el interior:

```
//CORRECTO
let name = "Bruce Wayne, a.k.a.: 'Batman'";
            // Fuera -> " "            Dentro -> ' '
let name = 'Bruce Wayne, a.k.a.: "Batman"';
            // Fuera -> ' '            Dentro -> " "
//INCORRECTO
let name = "Bruce Wayne, a.k.a.: "Batman"";
        // Fuera -> " "              Dentro -> " "
let name = 'Bruce Wayne, a.k.a.: 'Batman'';
        // Fuera -> ' '              Dentro -> ' '
```

Al igual que en los arrays, ya que los **strings** son cadenas de caracteres, podemos acceder a cada uno de ellos de manera individual haciendo uso de los corchetes:

```
const team = "Rocket";

console.log(team[1];     // Retorna "o"
```

A diferencia de los arrays, con la notación entre corchetes solo podemos acceder a los caracteres, pero no podemos eliminar o sustituir el valor de cada una de las posiciones del string original.

Métodos

A continuación os dejamos algunos métodos, realmente los más usados, que podemos usar cuando trabamos con strings.

length: El método **length** nos permite conocer la longitud de un string, es decir, el número de caracteres que lo conforman.

```
const country = "Italy";
console.log(country.length); // Retorna 5
```

includes: El método **includes** nos permite conocer si un caracter o una porción de caracteres se encuentran dentro de un string, devolviendo un valor booleano según se cumpla o no.

```
const quote = "To infinity and beyond";

const word = "infinity";
```

```
console.log(quote.includes(word)); // Retorna true
```

repeat: El método **repeat** devuelve un nuevo string con el número de copias del string donde lo estemos aplicando. Este número se le pasará por argumento al método.

```
const droid = "roger ";

console.log(droid.repeat(2)); // Retorna "roger roger";
```

replace: El método **replace** devuelve un string con el string insertado por argumento sustituido por el segundo argumento, es decir, como primer argumento recibirá el substring que queremos sustituir y como segundo argumento el substring que queremos que reemplace a este.

```
const movie = "Star Trek";

console.log(movie.replace("Trek", "Wars"));
// Retorna "Star Wars"
```

replaceAll: Este método funciona exactamente igual que el método **replace** pero, a diferencia del primero, este reemplaza todas las coincidencias encontradas en el strings con el substring indicado.

```
const quote = "Un Anillo para gobernarlos a todos. Un Anillo pa

console.log(quote.replaceAll("Anillo", "Gato"));
// Retorna "Un Gato para gobernarlos a todos.
// Un Anillo para encontrarlos, un Anillo para atraerlos
```

```
// a todos y atarlos en las tinieblas en la Tierra de Mordor
// donde se extienden las Sombras."
```

slice: El método **slice** devuelve una nueva cadena con la porción delimitada entre la posición del primer argumento y la posición del segundo argumento.

```
const album = "Master of Puppets";

console.log(album.slice(10, 13));
// Retorna "Pup"
```

split: El método **split** genera un array de tantos elementos como se indique en el segundo argumento indicándole el elemento separador en el primer argumento.

```
const phrase = "Buenos días, ¿cómo estás?";
console.log(phrase.split(" ", 3));
// Retorna ["Buenos","días",",¿cómo" ]
```

toLowerCase: El método **toLowerCase** devuelve el valor del string convertido a minúsculas.

```
const name = "Peter";
console.log(name.toLowerCase());
// Retorna "peter"
```

toUpperCase: El método **toUpperCase** devuelve el valor del string convertido a mayúsculas.

```
const name = "Peter";
console.log(name.toUpperCase());
```

```
// Retorna "PETER"
```

trim: El método **trim** elimina los espacios en blanco desde el principio hasta el final del string.

```
const quote = "    Luke, yo soy tu padre     ";
console.log(quote.trim());
// Retorna "Luke, yo soy tu padre"
```