Objeto Math
Métodos de Cadenas

Math

- Como vimos la clase pasada, Math es un objeto que tiene propiedades y métodos estáticos para resolver cálculos matemáticos.

- Math.random()

- Con el método random() generamos un número aleatorio entre 0 y 1

Número aleatorio

- Si queremos generar un numero entre un mínimo y un máximo, podemos aplicar el siguiente algoritmo:

- Math.floor((Math.random() * (max - min + 1)) + min);

- Math.floor(): Devuelve el máximo entero menor o igual a un número.

Función que retorne un numero aleatorio en un intervalo

- Podemos programar una función como la siguiente:
-

```
> const random = (min, max) => Math.floor((Math.random() * (max - min + 1)) + min)
<· undefined
```

Métodos de los Strings

- Así como los arrays, y todos los objetos que utlicemos de JavaScript, las cadenas también tienen métodos y propiedades.

- Length:

- “Hola”.length // nos devuelve un 4 (el tamaño de la cadena)

Acceder a un carácter de una cadena

Podemos hacerlo de dos formas:

- Usando el método charAt():

  “Hola”.charAt(1) // devuelve “o”

- Usando los corchetes e indicando el indice:

- “Hola”[1] // devuelve “o”

concat()

- Este método nos sirve para concatenar dos o mas cadenas en una sola.
- El método retorna una nueva cadena, sin modificar ninguna otra cadena.
- Ej:

```
let name1 = “Juan”;

let name2 = “Pedro”;
```

- `const fullName = name1.concat(“ ”, name2)`

includes()

- Retorna true si una cadena está incluida en otra, o false en caso contrario.

  Ejemplo:

- const texto = “Mi nombre es Felipe”
- const palabra = “nombre”

- if (texto.includes(palabra)) {

  console.log(“La palabra nombre está en la frase”)

- }

Métodos startsWith() y endsWith()

- Estos métodos nos devuelven true o false, si una cadena empieza/termina con una cadena dada como parámetro

- “Tengo 12 años”.startsWith(“Tengo”) // true

  “Tengo 12 años”.endsWith(“s”) // true

- “Tengo 12 años”.endsWith(“Años”) // false
- "Tengo 12 años".startsWith("Ta") // false

split()

- Divide una cadena en un array de cadenas, mediante la separación de la cadena en subcadenas.
- Recibe como parámetro una cadena a usar para la separación

Ejemplo:

```
const texto = “Hola como estas?”
```

- texto.split(“ ”) // [“Hola”, “como”, “estas?”]

Otro ejemplo del método split()

```
> const nombres = "Juan,Maria,Sofia,Ana,Pedro,Micaela"
< undefined
> const arrayNombres = nombres.split(",")
< undefined
> arrayNombres
< ▶ (6) ["Juan", "Maria", "Sofia", "Ana", "Pedro", "Micaela"]
>
```

indexOf()

- Este método nos retorna la posición en la que se encuentre una subcadena, dentro de una cadena.
- En caso de no encontrarla, nos devuelve -1
- Recibe dos parámetros, la subcadena que queremos buscar, y la posición desde la que queremos empezar a buscar.
- Este ultimo parámetro es opcional.

Ejemplos de String.indexOf()

```
> "Ballena Azul".indexOf("Azul")
< 8
> "Hola".indexOf("H")
< 0
> "Ballena Azul".indexOf("Azul", 8)
< 8
> "Ballena Azul".indexOf("Azul", 10)
< -1
```

trim()

- Con este método podemos eliminar los espacios en blanco, en los extremos de una cadena.

```
> let texto = "   Holaa     ";
< undefined
> texto.trim();
< "Holaa"
> let otroTexto = "Adiosss   ";
< undefined
> otroTexto.trim();
< "Adiosss"
> let mensaje = "  como estas?"
< undefined
> mensaje.trim();
< "como estas?"
>
```

toLowerCase() y toUpperCase()

- Transforma una cadena a minúsculas y mayúsculas, respectivamente.

```
> texto = "Hola como estas?"
"Hola como estas?"
> let textoEnMayus = texto.toUpperCase();
undefined
> textoEnMayus
"HOLA COMO ESTAS?"
> textoEnMayus.toLowerCase();
"hola como estas?"
>
```

substring()

- Extrae una subcadena, entre un intervalo de índices.

```
> texto = "Hoy es Viernes y hace frio";
"Hoy es Viernes y hace frio"
> texto.substring(4, 16)
"es Viernes y"
>
```

replace()

- Busca una subcadena dentro de una cadena, y la reemplaza por otra.
- Sólo reemplaza la primer coincidencia
- Este método no modifica el string, retorna un nuevo string

```
> "perro perro perro perro".replace("perro", "gato")
"gato perro perro perro"
> "Hoy es lunes y hace frio".replace("lunes", "viernes")
"Hoy es viernes y hace frio"
>
```

replaceAll()

- Se comporta casi igual que el método **replace**, salvo que reemplaza todas las coincidencias (no solo la primera)

```
> text = "Juan tiene 12 años, Juan debe ir a dormir, Juan es un niño travieso";
"Juan tiene 12 años, Juan debe ir a dormir, Juan es un niño travieso"
> text.replaceAll("Juan", "Pedro");
"Pedro tiene 12 años, Pedro debe ir a dormir, Pedro es un niño travieso"
>
```