Recorrer o Iterar un Array

Repasando...

- Para crear un array con tres nombres:

  ```
  const nombres = [“Juan”, “Pepe”, “Ana”];    // Una forma de hacerlo
  ```

- Para acceder al nombre “Juan”: nombres[0];
- Para acceder al nombre “Pepe”: nombres[1];
- Para acceder al nombre “Ana”: nombres[2];

- Este array tiene tamaño 3 (length)
- Sus índices van del 0 al 2.

Recorrer un Array

```
const nombres = [“Juan”, “Pepe”, “Ana”];
const index = 2;

nombres[index];
```

¿Qué nos devuelve la línea de arriba?

Iterar un Array con For

- Una de las formas de iterar en una colección de datos como lo son los arrays, es a través de la estructura for.

```
> const nombres = ["Juan", "Pepe", "Ana"];
<· undefined
> for (let index = 0; index < nombres.length; index++) {
      console.log(nombres[index]);
  }
  Juan
  Pepe
  Ana
<· undefined
>
```

Iterar un Array con For

```
const nombres = ["Juan", "Pepe", "Ana"];

for (let i = 0; i < nombres.length; i++)
{
    console.log("Posición " + i + " = " + nombres[i]);
}
```

Salida por consola:

```
2
3   const nombres = ["Juan", "Pepe", "Ana"];
4
5   for (let i = 0; i < nombres.length; i++)
6   {
7       console.log("Posición " + i + " = " + nombres[i]);
8   }
9
```

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL

```
joaquin@ubuntu:~/Documentos/clases_programacion_basica/codigo$ node do
Posición 0 = Juan
Posición 1 = Pepe
Posición 2 = Ana
joaquin@ubuntu:~/Documentos/clases_programacion_basica/codigo$
```

Asignar valores a un Array:

```
let nombres = new Array(5);

for (let i = 0; i < nombres.length; i++)
{
    const entrada = prompt("Ingrese un nombre para guardar:");
    nombres[i] = entrada;
}
```

Otra forma:

- Analicemos el siguiente código:

```
let frutas = [];    // declaro un array vacío

// el for se repetirá 5 veces
for (let i = 0; i < 5; i++)
{
    const fruta = prompt("Ingrese una fruta");
    frutas[i] = fruta;
}
```

- ¿Cual es el tamaño final del array?
- ¿Cual es el índice máximo del array?

Otras formas de recorrer un Array: For Of

- Sintaxis:
- ```
  for (let variableAuxiliar of array) {
      // Bloque del for of
  }
  ```

- Este for se utiliza para iterar objetos o colecciónes.
- Ejecutará un bloque de código para cada elemento de lo que estemos iterando.
  ```

For Of

```
let frutas = ["Manzana", "Kiwi", "Banana"];

for (const elem of frutas) {
    console.log(elem);
}
```

NOTA: En caso que la variable de iteración no se modifique, podemos usar const en vez de let

For In

- El for in se utiliza para iterar sobre propiedades numerables de los objetos o arrays, como puede ser el índice.
- Ejemplo

```
> const nombres = ["Juan", "Pepe", "Ana"];
<· undefined
> for (const index in nombres) {
   console.log(nombres[index]);
  }
  Juan
  Pepe
  Ana
```

Analicemos el siguiente código...

```
let numeros = [10, 20, 30];

console.log(numeros.toString());

for (const index in numeros) {
    numeros[index]++;
}

console.log(numeros.toString());
```

Otra forma de recorrer Arrays: For Each

- El método forEach() ejecuta la función indicada una vez por cada elemento del array

```
const array1 = ['a', 'b', 'c'];

array1.forEach(element => console.log(element));

// expected output: "a"
// expected output: "b"
// expected output: "c"
```

El método forEach recibe como **parámetro** una **función** que se ejecutará por cada elemento del array.

A estas funciónes que se pueden recibir como parámetro se les llama **callbacks**

Array.map()

- El método map() crea un nuevo array con los resultados de la llamada a la función indicada (callback) aplicados a cada uno de sus elementos.
- Otro uso de esta función, es para recorrer el array.
-

```
const numeros = [10, 20, 30];

const nuevoArray = numeros.map(function(num) {
    return num += 1;
});

// Otra forma de escribir la funcion de arriba:

const arr = numeros.map(elem => ++elem );

console.log(nuevoArray.join());
console.log(arr.join());
```

Analicemos este código

En resumen, para recorrer Arrays:

- For clásico
- For in
- For of
- For each
- Map (*)
- Otras estructuras de repetición, como while y do while

Próxima clase:

- Veremos mas propiedades de los arrays
- Haremos ejercicios
- Empezaremos con los arrays asociativos.