iesperemariaorts

Desarrollo web en entorno cliente

Tema 2

Javascript

Colecciones

iespereemariaorts

Colecciones: Arrays

- Son de tipo objeto -> instanciamos clase "Array"
- Constructor recibe:
* 0 parámetros: Array vacío
* Número: Tamaño del array
* Elementos recibidos.

```
let a = new Array(); // Crea un array vacío
a[0] = 13;
console.log(a.length); // Imprime 1
console.log(a[0]); // Imprime 13
console.log(a[1]); // Imprime undefined
```

```
let a = new Array(12); // Crea un array de tamaño 12
console.log(a.length); // Imprime 12
a[20] = "Hello";
console.log(a.length); // Ahora imprime 21 (0-20). Las posiciones 0-19 tendrán el valor undefined
```

```
let a = ["a", "b", "c", "d", "e"]; // Array de tamaño 5, con 5 valores inicialmente
console.log(typeof a); // Imprime object
console.log(a instanceof Array); // Imprime true. a es una instancia de array
a[a.length] = "f"; // Insertamos in nuevo elemento al final
```

iesperemariaorts

Colecciones: Recorriendo Arrays

- Con bucles while y for, recorriendo con el índice.

```
let  ar = new Array(4, 21, 33, 24, 8);

let i = 0;
while(i < ar.length) { // Imprime 4 21 33 24 8
   console.log(ar[i]);
   i++;
}

for(let i = 0; i < ar.length; i++) { // Imprime 4 21 33 24 8
   console.log(ar[i]);
}

for (let i in ar) { // Imprime 4 21 33 24 8
   console.log(ar[i]);
}
```

iesperemariaorts

Colecciones: Recorriendo Arrays

- Desde ES2015: iterar sin utilizar el índice:

```
let a = ["Item1", "Item2", "Item3", "Item4"];

for(let index in a) {
   console.log(a[index]);
}
```

```
for(let item of a) { // Hace lo mismo que el bucle anterior
   console.log(item);
}
```

```
let str = "abcdefg";

for(let letter of str) {
if(letter.match(/^[aeiou]$/)) {
   console.log(letter + " es una vocal");
} else {
   console.log(letter + " es una consonante");
}
```

iesperemariaorts

Colecciones: Métodos Arrays

-Insertar valores al principio de un array (unshift) y al final (push)

```
let   a = [];
a.push("a"); // Inserta el valor al final del array
a.push("b", "c", "d"); // Inserta estos nuevos valores al final
console.log(a); // Imprime ["a", "b", "c", "d"]
a.unshift("A", "B", "C"); // Inserta nuevos valores al principio del array
console.log(a); // Imprime ["A", "B", "C", "a", "b", "c", "d"]
```

- Eliminar del principio (shift) y del final (pop) del array

```
console.log(a.pop()); // Imprime y elimina la última posición → "d"
console.log(a.shift()); // Imprime y elimina la primera posición → "A"
console.log(a); // Imprime ["B", "C", "a", "b", "c"]
```

iesperemariaorts

Colecciones: Métodos Arrays

- join():devuelve un string con todos los elementos separados por coma

```
let  a = [3, 21, 15, 61, 9];
console.log(a.join()); // Imprime "3,21,15,61,9""
console.log(a.join(" -#- ")); // Imprime "3 -#- 21 -#- 15 -#- 61 -#- 9"
```

- concat: concatena dos arravs

```
let  a = ["a", "b", "c"];
let  b = ["d", "e", "f"];
let  c = a.concat(b);
console.log(c); // Imprime ["a", "b", "c", "d", "e", "f"]
console.log(a); // Imprime ["a", "b", "c"] . El array original no ha sido modificado
```

- slice: devuelve sub-array

```
let a = ["a", "b", "c", "d", "e", "f"];
let b = a.slice(1, 3); // (posición de inicio → incluida, posición final → excluida)
console.log(b); // Imprime ["b", "c"]
console.log(a); // Imprime ["a", "b", "c", "d", "e", "f"]. El array original no es modificado
console.log(a.slice(3)); // Un parámetro. Devuelve desde la posición 3 al final → ["d", "e", "f"]
```

iesperemariaorts

Colecciones: Métodos Arrays

- splice: elimina elementos y devuelve los eliminados

```
let  a = ["a", "b", "c", "d", "e", "f"];
a.splice(1, 3); // Elimina 3 elementos desde la posición 1 ("b", "c", "d")
console.log(a); // Imprime ["a", "e", "f"]
```

- reverse: invierte el orden

- sort: ordena elementos array (los trata como strings)

```
let a = [20, 6, 100, 51, 28, 9];
a.sort(); // Ordena el array original
console.log(a); // Imprime [100, 20, 28, 51, 6, 9]
a.sort((n1, n2) => n1 - n2);
console.log(a); // Imprime [6, 9, 20, 28, 51, 100]
```

- indexOf: podemos conocer si el valor que le pasamos se encuentra en el array o no

```
let a = [3, 21, 15, 61, 9, 15];
console.log(a.indexOf(15)); // Imprime 2
console.log(a.indexOf(56)); // Imprime -1. No encontrado
console.log(a.lastIndexOf(15)); // Imprime 5
```

iesperemarjaorts

Colecciones: Métodos Arrays

- every: devolverá un boolean indicando si todos los elementos del array cumplen cierta condición.

```
let a = [3, 21, 15, 61, 9, 54];
console.log(a.every(num => num < 100));  // Comprueba si cada número es menor a 100. Imprime true
console.log(a.every(num => num % 2 == 0)); // Comprueba si cada número es par. Imprime false
```

- forEach: iterar por los elementos de un array

```
let a = [3, 21, 15, 61, 9, 54];
let sum = 0;
a.forEach(num => sum += num);
console.log(sum); // Imprime 163
```

- filter: devuelve array que contenga sólo los elementos que cumplan cierta condición

```
let a = [4, 21, 33, 12, 9, 54];
console.log(a.filter(num => num % 2 == 0));  // Imprime [4, 12, 54]
```

iesperemariaorts

Extensiones de Array en ES 2015

- Array.of(value): Instanciar un array con un solo valor

```
let array = new Array(10); // Array vacío (longitud 10)
let array = Array(10); // Mismo que arriba: array vacío ( longitud 10)
let array = Array.of(10); // Array con longitud 1 -> [10]
let array = [10]; // // Array con longitud 1 -> [10]
```

- Array.from(array, func): Crea array desde otro array

```
let array = [4, 5, 12, 21, 33];
let array2 = Array.from(array, n => n * 2);
console.log(array2); // [8, 10, 24, 42, 66]
```

- Array.fill(value) – Array.fill(value, start, end)

```
let sums = new Array(6); // Array con 6 posiciones
sums.fill(0); // Todas las posiciones se inicializan a 0
console.log(sums); // [0, 0, 0, 0, 0, 0]

let numbers = [2, 4, 6, 9];
numbers.fill(10); // Inicializamos las posiciones al valor 10
console.log(numbers); // [10, 10, 10, 10]
```

```
let numbers = [2, 4, 6, 9, 14, 16];
numbers.fill(10, 2, 5); // Las posiciones 2,3,4 se ponen a 10
console.log(numbers); // [2, 4, 10, 10, 10, 16]

let numbers2 = [2, 4, 6, 9, 14, 16];
numbers2.fill(10, -2); // Las dos últimas posiciones se ponen a 10
console.log(numbers2); // [2, 4, 6, 9, 10, 10]
```

iespereemariaorts

Colecciones: Rest - Spread

- rest: acción de transformar un grupo de parámetros en un array.

```
function getMedia(...notas) {
   console.log(notas); // Imprime [5, 7, 8.5, 6.75, 9] (está en un array)
   let total = notas.reduce((total,notas) => total + notas, 0);
   return total / notas.length;
}
console.log(getMedia(5, 7, 8.5, 6.75, 9)); // Imprime 7.25
```

- spread: extraer los elementos de un array (o de un string) a variables.

```
let nums = [12, 32, 6, 8, 23];
console.log(Math.max(nums)); // Imprime NaN (array no es válido), deben ser números
console.log(Math.max(...nums)); // Imprime 32 -> equivalente a Math.max(12, 32, 6 ,8 ,23)

let a = [1, 2, 3, 4];
let b = a; // Referencia el mismo array que 'a' (las modificaciones afectan a ambos).
let c = [...a]; // Nuevo array (copia de a) -> contiene [1, 2, 3, 4]
```

```
let a = [1, 2, 3, 4];
let b = [5, 6, 7, 8];
let c = [...a,...b, 9, 10]; // [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

iespereMariaorts

Colecciones: Desestructuración de arrays

- acción de extraer elementos individuales de un array directamente en variables individuales

```
let array = [150, 400, 780, 1500, 200];
let [first, second, third] = array; // Asigna los tres primeros elementos del array
console.log(third); // Imprime 780
```

```
let array = [150, 400, 780, 1500, 200];
let [first, , third] = array; // Asigna el primer y tercer elemento
console.log(third); // Imprime 780
```

```
let array = [150, 400, 780, 1500, 200];
let [first, second, ...rest] = array; // rest -> array
console.log(rest); // Imprime [780, 1500 ,200]
```

```
let array = ["Peter", "John"];
let [first, second = "Mary", third = "Ann"] = array; // rest -> array
console.log(second); // Imprime "John"
console.log(third); // Imprime "Ann" -> valor por defecto
```

iesperemariaorts

Colecciones: Map

- colección que guarda parejas de [clave,valor]. Un objeto puede ser map con limitaciones(strings y enteros como claves)

```
let obj = {
    0: "Hello",
    1: "World",
    prop1: "This is",
    prop2: "an object"
}

console.log(obj[1]); // Imprime "World"
console.log(obj["prop1"]); // Imprime "This is"
console.log(obj.prop2); // Imprime "an object"
```

iesperemariaorts

Colecciones: Map

- La nueva colección Map permite usar cualquier objeto como clave.

- Creamos la colección usando el constructor new Map().

- Usamos los métodos set, get y delete para almacenar, obtener o eliminar un valor basado en una clave.

iesperemariaorts

Colecciones: Map

```
let person1 = {name: "Peter", age: 21};
let person2 = {name: "Mary", age: 34};
let person3 = {name: "Louise", age: 17};

let hobbies = new Map(); // Almacenará una persona con un array de hobbies (string)
hobbies.set(person1, ["Tennis", "Computers", "Movies"]);
console.log(hobbies); // Map {Object {name: "Peter", age: 21} => ["Tennis", "Computers", "Movies"]}

hobbies.set(person2, ["Music", "Walking"]);
hobbies.set(person3, ["Boxing", "Eating", "Sleeping"]);
console.log(hobbies);
```

```
Map {Object {name: "Peter", age: 21} => ["Tennis", "Computers", "Movies"], Object
▼{name: "Mary", age: 34} => ["Music", "Walking"], Object {name: "Louise", age: 17}
=> ["Boxing", "Eating", "Sleeping"]} i
  size: (...)
  ▶ __proto__: Map
  ▼ [[Entries]]: Array[3]
    ▼ 0: {Object => Array[3]}
      ▼ key: Object
          age: 21
          name: "Peter"
        ▶ __proto__: Object
      ▼ value: Array[3]
          0: "Tennis"
          1: "Computers"
          2: "Movies"
          length: 3
```

iesperemariaorts

Colecciones: Map

Devuelve 2 posiciones: 0 "key" y 1 "value"

```
console.log(hobbies.size); // Imprime 3
hobbies.delete(person2); // Elimina person2 del Map
console.log(hobbies.size); // Imprime 2
console.log(hobbies.get(person3)[2]); // Imprime "Sleeping"

/** Imprime todo:
 * Peter:  Tennis, Computers, Movies
 * Louise: Boxing, Eating,  Sleeping */
for(let entry of hobbies) {
   console.log(entry[0].name + ": " + entry[1].join(", "));
}
for(let [person, hobArray] of hobbies) { // Mejor
   console.log(person.name + ": " + hobArray.join(", "));
}

hobbies.forEach((hobArray, person) => { // Mejor
   console.log(person.name + ": " + hobArray.join(", "));
});
```

```
let prods = [
   ["Computer", 345],
   ["Television", 299],
   ["Table", 65]
];

let prodMap = new Map(prods);
```

iespereMariaorts

Colecciones: Set

- Como Map pero no almacena la los valores, solo las claves.
- Colección que no permite valores duplicados.
- Usamos los métodos add, delete y has

```
let set = new Set();
   set.add("John");
   set.add("Mary");
   set.add("Peter");
   set.delete("Peter");
   console.log(set.size); // Imprime 2
   set.add("Mary"); // Mary ya existe
   console.log(set.size); // Imprime 2

   // Itera a través de los valores
   set.forEach(value => {
      console.log(value);
   })
```

```
let names = ["Jennifer", "Alex", "Tony", "Johny", "Alex", "Tony", "Alex"];
let nameSet = new Set(names);
console.log(nameSet); // Set  {"Jennifer", "Alex", "Tony", "Johny"}
```

iesperemariaorts

Funciones globales

- parseInt(value) → Transforma cualquier valor en un entero. Devuelve el valor entero, o NaN si no puede ser convertido.
- parseFloat(value) → Igual que parseInt, pero devuelve un decimal.
- IsNaN(value) → Devuelve true si el valor es NaN.
- isFinite(value) → Devuelve true si el valor es un número finito o false si es infinito.
- Number(value) → Transforma un valor en un número (o NaN).
- String(value) → Convierte un valor en un string (en objetos llama a toString()).

iesperemariaorts

Objeto Math

- round(x) → Redondea x al entero más cercano.
- floor(x) → Redondea x hacia abajo (5.99 → 5. Quita la parte decimal)
- ceil(x) → Redondea x hacia arriba (5.01 → 6)
- min(x1,x2,...) → Devuelve el número más bajo de los argumentos que se le pasan.
- max(x1,x2,...) → Devuelve el número más alto de los argumentos que se le pasan.
- pow(x, y) → Devuelve x^y (x elevado a y).
- abs(x) → Devuelve el valor absoluto de x.
- random() → Devuelve un número decimal aleatorio entre 0 y 1 (no incluidos).

```
console.log("Raíz cuadrada de 9: " + Math.sqrt(9));
console.log("El valor de PI es: " + Math.PI);
console.log(Math.round(4.546342));
// Número aleatorio entre 1 y 10
console.log(Math.floor(Math.random() * 10) + 1);
```

iesperemariaorts

Fechas

- Clase Date: encapsula información sobre fechas y métodos para operar.

```
let date = new Date(); // Crea objeto Date almacena la fecha actual
console.log(typeof date); // Imprime object
console.log(date instanceof Date); // Imprime true
console.log(date); // Imprime (en el momento de ejecución)
```

- Podemos mandar al constructor los milisegundos desde el 1/1/1970 (epoch – UNIX Time)

```
let date = new Date(1363754739620); // Nueva fecha 20/03/2013 05:45:39 (milisegundos desde Epoch)
let date2 = new Date(2015, 5, 17, 12, 30, 50); // 17/06/2015 12:30:50 (Mes empieza en 0 -> Ene, ... 11 -> Dic)
let date3 = new Date("2015-03-25"); // Formato de fecha largo sin la hora YYYY-MM-DD (00:00:00)
let date4 = new Date("2015-03-25T12:00:00"); // Formato fecha largo con la fecha
let date5 = new Date("03/25/2015"); // Formato corto MM/DD/YYYY
let date6 = new Date("25 Mar 2015"); // Formato corto con el mes en texto (March también sería válido).
let date7 = new Date("Wed Mar 25 2015 09:56:24 GMT+0100 (CET)"); // Formato completo con el timezone
```

```
let now = new Date();

console.log(now.toString());
console.log(now.toISOString()); // Imprime 2016-06-26T18:00:31.246Z
console.log(now.toUTCString()); // Imprime Sun, 26 Jun 2016 18:02:48 GMT
console.log(now.toDateString()); // Imprime Sun Jun 26 2016
```