



Arrays y otras colecciones indexadas



2º DAW - DWEC
Daniel Marín López
Laura Luque Bravo
Alejandro Vaquero Abad

START



ÍNDICE

1

¿Cuales son?



2

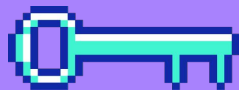
Tipos y Utilidad

3

Ejemplos



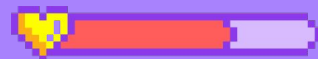
COLECCIONES CON CLAVE



Ordenan sus datos mediante una clave y son iterables en el orden de inserción: Map y Set.



COLECCIONES INDEXADAS



Las estructuras que organizan sus datos a través de un índice son conocidas como Array y TypedArray

1

¿Cuáles
son?



COLECCIONES INDEXADAS

Array

Definición

Son objetos que actúan como listas y proporcionan métodos tanto para efectuar operaciones de recorrido como para la mutación de datos. En un array, tanto la longitud como el tipo de los elementos son variables.



TypedArray

Definición

Objetos similares a los arrays pero con un mecanismo especial que maneja datos binarios sin procesar en el búffer y tipos de datos específicos. Contienen datos del mismo tipo.

ÍNDICE

1

¿Cuales son?

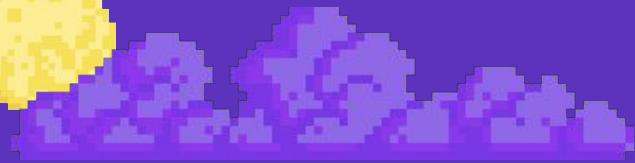

2

Tipos y Utilidad

3

Ejemplos





2

Tipos y utilidad: Arrays



ARRAYS

1

FUNCIÓN

Los arrays son estructuras que almacenan elementos de diversos tipos en una lista ordenada. Son eficientes para gestionar conjuntos de datos, permitiendo operaciones como adición, eliminación, búsqueda y modificación de elementos.

2

SINTAXIS

```
1 let vocales = ["a", "b", "c", "d", "e"]
```

3

CUANDO SE USAN

- Acceder a elementos por índice
- Agregar elementos al final del array
- Eliminar el último elemento del array
- Obtener la longitud del array
- Recorrer un array con bucles, como for o forEach.
- Realizar búsquedas en un array utilizando métodos como indexOf o find.



EJEMPLOS

1 Acceso a los elementos

FORMA	FUNCIÓN
<code>array.length</code>	Devuelve el nº de elementos de un array
<code>operador []</code>	Posición del elemento deseado en el array

EJEMPLOS

1 Acceso a los elementos

```
//Array normal
```

```
let alumnos = ["Daniel Marín", "Laura Luque", "Alejandro Vaquero"];  
console.log(alumnos); //['Daniel Marín', 'Laura Luque', 'Alejandro Vaquero']  
console.log(alumnos.length); // 3
```

```
let long = new Array(5);  
console.log(long); //(5) [empty × 5]  
console.log(long.length); // 5
```

EJEMPLOS

1 Acceso a los elementos

```
//Índices en un array
let characters = ["Reimu Hakurei", "Marisa Kirisame", , "Sakuya Izayoi", , , , "Sanae Kochiy
console.log(characters);
console.log(characters[0]);
console.log(characters[2]);
console.log(characters[3]);
/*
characters[0] = "Reimu Hakurei"
characters[2] = undefined
characters[3] = "Sakuya Izayoi"
characters[5] = undefined
*/
```

EJEMPLOS

2 Añadir/eliminar elementos

FORMA	FUNCIÓN	USOS	RENDIMI.
<code>array.unshift()</code>	Añade elementos al principio del array y devuelve su tamaño	COLA	LENTO
<code>array.shift()</code>	Elimina el primer elemento del array, lo devuelve y el resto avanzan una posición	COLA	LENTO
<code>array.push()</code>	Añade elementos al final del array y devuelve su tamaño	PILA	RAPIDO
<code>array.pop()</code>	Elimina el último elemento del array y lo devuelve	PILA	RAPIDO

EJEMPLOS

2 Añadir/eliminar elementos

```
//Añadir y eliminar valores de un array (shift, unshift)  
let films = ["Brother Bear", "Insidious", "IT"];  
console.log(films.shift()); // 'Brother Bear'  
console.log(films); // (2) ['Insidious', 'IT']  
films.unshift("Brother Bear");  
console.log(films); // (3) ['Brother Bear', 'Insidious', 'I  
T']
```

EJEMPLOS

2 Añadir/eliminar elementos

```
//Añadir y eliminar valores de un array (push, pop)
let games = ["東方鬼形獣 ～ Wily Beast and Weakest Creature", "東方虹龍洞 ～ Unconnected Marketeer"];
games.push("東方獸王園 ～ Unfinished Dream of All Living Ghost");
console.log(games);
// (3) ['東方鬼形獣 ～ Wily Beast and Weakest Creature', '東方虹龍洞 ～ Unconnected Marketeers', '東方獸王園 ～ Unfinished Dream of All Living Ghost']

games.pop();
console.log(games);
// (2) ['東方鬼形獣 ～ Wily Beast and Weakest Creature', '東方虹龍洞 ～ Unconnected Marketeers']
```

EJEMPLOS

3 Métodos estáticos

FORMA	FUNCIÓN
<code>Array.of()</code>	Crea un array con el n° pasado por parámetro
<code>Array.from()</code>	Crea un array a partir de un objeto iterable
<code>Array.isArray()</code>	Devuelve true si es un Array y false si no lo es
<code>Array.fill()</code>	Modifica un array rellenandolo con un valor entre un índice inicio y un índice fin.

EJEMPLOS

3 Métodos estáticos

```
//Array.of
let comida = Array.of("Salmón", "Pizza", "Solomillo");
console.log(comida); // (3) ['Salmón', 'Pizza', 'Solomillo']
```

```
//Array.from
let frase = Array.from("Lourdes la mejor");
console.log(frase); // (16) ['L', 'o', 'u', 'r', 'd', 'e', 's', ' ', 'L', 'a', ' ', 'm', 'e', 'j', 'o', 'r']
let num1 = Array.from(2);
console.log(num1); // []
let pot = Array.from([1, 4, 8], x => x**2);
console.log(pot); // (3) [1, 16, 64]
```


EJEMPLOS

3 Métodos estáticos

```
//Array.isArray
console.log(`¿Es iterable alumnos? ${Array.isArray(alumnos) ? "Sí" : "No"}`); // Sí
console.log(`¿Es iterable un objeto ({})? ${Array.isArray({}) ? "Sí" : "No"}`); // No
console.log(`¿Es iterable asignaturas (new Array)? ${Array.isArray(asignaturas) ? "Sí" : "No"}`); // Sí
console.log(`¿Es iterable "kk"? ${Array.isArray("kk") ? "Sí" : "No"}`); // No
console.log(`¿Es iterable Array.prototype? ${Array.isArray(Array.prototype) ? "Sí" : "No"}`); // Sí
console.log(`¿Es iterable {_proto_:Array.prototype}? ${Array.isArray({_proto_:Array.prototype}) ? "Sí" : "No"}`); // N
o
```

```
console.log("Array.fill()");
let relleno1 = [1, 2, 3, 4];
console.log(relleno1.fill(7)); // (4) [7, 7, 7, 7]
console.log(relleno1.fill("kk", 2)); // (4) [7, 7, 'kk', 'kk']
let relleno2 = [1, 2, 3, 4];
console.log(relleno2.fill("kk", 1, 3)); // (4) [1, 'kk', 'kk', 4]
```

EJEMPLOS

4 Métodos de comprobación

FORMA	FUNCIÓN
<code>array.every()</code>	Determina si todos los elementos de un array satisfacen una condición
<code>array.find()</code>	Devuelve el primer elemento que cumple una condición dada
<code>array.indexOf()</code>	Devuelve el índice de un elemento dado, -1 si no lo encuentra
<code>array.lastIndexOf()</code>	Devuelve el último índice de un elemento dado, -1 si no lo encuentra
<code>array.join()</code>	Une los elementos de una array a una cadena y la devuelve, también se puede pasar como parámetro un elemento que los concatene
<code>array.includes()</code>	Devuelve true si el array incluye el elemento dado
<code>array.at()</code>	Devuelve el elemento en ese índice. Se permiten números positivos y negativos, siendo estos últimos el final del array
<code>array.filter()</code>	Devuelve un array de elementos de otro array cumplen una determinada condición.

EJEMPLOS

4 Métodos de comprobación

```
let notas = [7, 9, 4, 8, 6];  
console.log(notas);  
  
//Array.every  
let aprobado = (valor) => valor >= 5;  
console.log(notas.every(aprobado)); // false  
e
```

```
//Array.find  
let found1 = notas.find((valor) => valor == 4);  
console.log(found1); // 4  
  
let found2 = notas.find((valor) => valor >= 5);  
console.log(found2); // undefined
```

EJEMPLOS

4 Métodos de comprobación

```
//Array.indexOf  
let frutas = ["banana", "pera", "naranja", "melón", "sandía"];  
  
console.log(frutas.indexOf("pera")); // 1  
console.log(frutas.indexOf("sandía")); // 4  
console.log(frutas.indexOf("melocotón")); // -1
```

```
let moda1 = ["rojo", "azul", "rojo", "rojo"];  
console.log(modas1);  
  
console.log("Array.lastIndexOf");  
console.log(modas1.lastIndexOf("rojo")); // 3  
console.log(modas1.lastIndexOf("azul")); // 1
```

EJEMPLOS

4 Métodos de comprobación

```
console.log("Array.includes");  
console.log(frutas.includes("naranja")); //true  
console.log(frutas.includes("mandarina")); //false
```

```
console.log("Array.join");  
console.log(frutas.join()); // banana,pera,naranja,melón,sandía  
console.log(frutas.join(" - ")); // banana - pera - naranja - melón - sandía
```

EJEMPLOS

4 Métodos de comprobación

```
//Array.at  
console.log(frutas.at(2)); // naranja  
console.log(frutas.at(-1)); // sandía  
console.log(frutas.at(3)); // melón  
console.log(frutas.at(7)); // undefined
```

```
console.log("Array.filter");  
console.log(notas.filter((value) => value >= 5)); // (4) [7, 9, 8, 6]  
console.log(frutas.filter((cad) => cad.length >= 5)); // (4) ['banana', 'naranja', 'melón', 'sandía']
```

EJEMPLOS

5 Métodos de modificación

FORMA	FUNCIÓN
<code>array.slice()</code>	Devuelve una copia de una parte de un array y la mete en un nuevo array empezando por el inicio. El array original no es modificado.
<code>array.splice()</code>	Cambia el contenido del array añadiendo y/o eliminando elementos a un array.
<code>array.split()</code>	Divide una cadena en un array de cadenas mediante la separación en subcadenas.
<code>array.concat()</code>	Une varios arrays en uno solo sin modificar el contenido y lo devuelve.

EJEMPLOS

5 Métodos de modificación

```
console.log("array.slice");  
console.log(asignaturas.slice(1, 4)); // ['DWES', 'DeAW', 'DIW', 'EIE']
```

```
console.log("array.splice");  
characters.splice(2,0,"Youmu Konpaku");  
console.log(characters); // ['Reimu Hakurei', 'Marisa Kirisame', 'Youmu Konpaku',  
//empty, 'Sakuya Izayoi', empty × 3, 'Sanae Kochiya']
```


EJEMPLOS

5 Métodos de modificación

```
console.log("array.split");  
let cadena1 = "Buenos Días";  
console.log(cadena1);  
let subarray = cadena1.split("", 4);  
console.log(subarray); // (4) ['B', 'u', 'e', 'n']
```

```
console.log("array.concat");  
console.log(comida.concat(frutas)); // (8) ['Salmón', 'Pizza', 'Solomillo',  
// 'banana', 'pera', 'naranja', 'melón', 'sandía']
```

EJEMPLOS

6 Métodos de ordenación

FORMA	FUNCIÓN
<code>array.sort()</code>	Devuelve un array dado, por defecto se ordena en base a la posición del valor string de acuerdo a su valor Unicode.
<code>array.reverse()</code>	Invierte el orden del array.

EJEMPLOS

6 Métodos de ordenación

```
let cosas = ["lourdes", "Lourdes", "1 Lourdes", "2 Lourdes", "lourdes 3"];  
console.log("array.sort");  
console.log(frutas.sort()); // (5) ['banana', 'melón', 'naranja', 'pera', 'sandía']  
console.log(notas.sort()); // (5) [4, 6, 7, 8, 9]  
console.log(cosas.sort()); // (5) ['1 Lourdes', '2 Lourdes', 'Lourdes', 'Lourdes', 'lourdes 3']
```

```
console.log("array.reverse");  
console.log(frutas.reverse()); // (5) ['sandía', 'pera', 'naranja', 'melón', 'banana']
```

EJEMPLOS

7 Formas de recorrer un array

For clásico

Utiliza un índice que incrementa hasta llegar al final del array

For in

A menudo usado para objetos pero que también pueden usar los arrays

For of

Utiliza un iterador constante del array

Foreach

El más habitual ya que es un método del array que da el elemento en sí



EJEMPLOS

7 Formas de recorrer un array

```
console.log("For clásico");  
for (let i = 0; i < games.length; i++) {  
  const element = games[i];  
  console.log(element);  
}
```

//東方鬼形獸 ~ *Wily Beast and Weakest Creature*

//東方虹龍洞 ~ *Unconnected Marketeers*

//東方獸王園 ~ *Unfinished Dream of ALL Living Ghost*

```
console.log("For of");  
for (const element of games) {  
  console.log(element);  
}
```

EJEMPLOS

7 Formas de recorrer un array

```
console.log("For in");
for (const key in films) {
  if (Object.hasOwnProperty.call(films, key)) {
    const element = films[key];
    console.log(element);
  }
}
//Brother Bear
//Insidious
//IT
```

```
console.log("Foreach");
films.forEach(element => {
  console.log(element);
});
```



Tipos y utilidad:

- **Typed Arrays**



TYPED ARRAYS

1

FUNCIÓN

Los `TypedArrays` son arrays que almacenan datos en formato binario y tienen un tamaño fijo. Son útiles para trabajar con datos binarios como imágenes, sonidos, etc.

2

SINTAXIS

```
const typedArray = new TypedArrayConstructor(length);  
  
const intArray = new Int16Array(5);
```

3

CUANDO SE USAN

Cuando necesitas trabajar con datos binarios o cuando deseas realizar operaciones numéricas eficientes en un conjunto de datos. Están diseñados específicamente para manejar datos en formato binario y proporcionan un control preciso sobre el tipo de datos almacenados en cada elemento del array.



TYPED

ARRAYS

Búfer

Implementado por el objeto `ArrayBuffer` el cual representa un búfer de datos binarios genéricos de longitud fija.

Vistas

Necesarias para acceder a la memoria que contiene el búfer. Representa el búfer en un formato específico, y lo usa para leer y escribir el contenido del búfer.



TYPED ARRAYS

1 Búfer

ArrayBuffer (16 bytes)																
UInt8Array	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
UInt16Array	0		1		2		3		4		5		6		7	
UInt32Array	0				1				2				3			
Float64Array	0								1							

EJEMPLOS

1 Búfer

```
const buffer = new ArrayBuffer(16);
```

TYPED ARRAYS

2 Vistas

TIPO	INTERVALO	DESCRIPCIÓN	TIPO DE IDL WEB
Int8Array	-128 a 127	Dos enteros complementarios de 8 bits con signo.	byte
Uint8Array	0 a 255	Entero de 8-bit sin signo	octal
Uint8ClampedArray	0 a 255	Entero de 8 bits sin signo (sujeito)	octal
Int16Array	-32768 a 32767	Dos enteros complementarios de 16 bits con signo	short
Int32Array	-2147483648 a 2147483647	Dos enteros complementarios de 32 bits con signo	long
Float32Array	1.2×10^{-38} a 3.4×10^{38}	Número de coma flotante IEEE de 32 bits (7 dígitos significativos, p. ej., 1.1234567)	float sin restricciones
BigInt64Array	-2^{63} a $2^{63}-1$	Dos enteros complementarios de 64 bits con signo	bigint

EJEMPLOS

2 Vistas

```
const buffer = new ArrayBuffer(8);

const int32Array = new Int32Array(buffer);

int32Array[0] = 42;
int32Array[1] = 77;


console.log("Elemento 1:", int32Array[0]);
console.log("Elemento 2:", int32Array[1]);
```



WEBGRAFÍA

- ❑ https://developer.mozilla.org/es/docs/Web/JavaScript/Reference/Global_Objects/Array
- ❑ http://chuwiki.chuidiang.org/index.php?title=Arrays_y_Objetos_en_JavaScript
- ❑ <https://docs.oracle.com/middleware/1213/adf/api-reference-javascript-faces/org/ecmascript/object/Array.html>
- ❑ <https://javascript.info/array#performance>
- ❑ https://developer.mozilla.org/es/docs/Web/JavaScript/Guide/Indexed_collections
- ❑ https://developer.mozilla.org/es/docs/Web/JavaScript/Guide/Indexed_collections
- ❑ <http://blog.enriqueoriol.com/2016/04/novedades-es6-typed-arrays.html>
- ❑ <https://lenguajejs.com/javascript/fundamentos/arrays/>





¿Alguna
pregunta?



A pixel art illustration of a town square. In the center is a large, light blue rectangular sign with a dark blue border. To the left is a shop with a green and white striped awning and a sign that says 'BOUTIQUE'. A blonde woman in a red dress stands next to a green bush in front of the shop. To the right of the sign, a small white cat with a red bow sits on a ledge. In the bottom right corner, there is a pink fire hydrant and a large, multi-layered cake on a stand. The background features a blue sky with a large yellow sun, a blue mountain, and a blue cloud. The ground is a grid of blue and green squares.

**GRACIAS POR
SU ATENCIÓN**