Destructuring

Destructuring

- El **destructuring** es una técnica que nos permite asignar valores a variables desde un objeto más complejo, o desde arrays con varios elementos, mediante la técnica de "object pattern"

Destructuring

Destructuring Target <= Destructuring Source

Siendo Target y Source alguno de estos dos:

- Object
- Array

Destructuring de objects

```
const obj = { name: ‘Emiliano', nick: ‘Oke' };
const {name: myName, nick: myNick} = obj;
```

El valor final de **myName** será “Emiliano” y el valor final de **myNick** será “Oke”

Simplificando propiedades

{name}

es la forma abreviada de

{name : name}

Destructuring de objects con propiedades simplificadas

```
const obj = { name: ‘Emiliano', nick: ‘Oke' };
const {name, nick} = obj;
```

El resultado es el mismo que el anterior.

El valor final de **name** será “Emiliano” y el valor final de **nick** será “Oke”

Array destructuring

```
const myArray = ['a', 'b'];
const [x, y] = myArray;
```

El valor final de x será “a” y el valor de y será “b”

El destructuring puede utilizarse en combinación con las asignaciones con "**const**", "**let**", y "**var**"

Destructuring de objetos complejos

El destructuring se puede anidar, de manera que se puede hacer destructuring de objetos con estructuras anidadas o de arrays que tienen arrays como elementos

Sólo lo necesario

Cuando hacemos destructuring sobre un objeto, no es necesario que "tomemos" todas sus propiedades, sólo las que necesitamos

```
const source = { x: 7, y: 3 };
const { x } = source;
```

En el caso de destructuring de arrays es lo mismo, podemos tomar los elementos del array que sean necesarios

Propiedades por defecto

También se pueden establecer valores por defecto a las propiedades

```
const { x, y = 1 } = {}; // x = undefined; y = 1
```

Elision

- En el destructuring de arrays se puede utilizar "elision"
- De esa forma se pueden omitir uno o más elementos de determinada posición del array

const [,, x, y] = ['a', 'b', 'c', 'd']; *// x = 'c'; y = 'd‘ y no se utilizan los primeros valores “a” y “b”*

rest operator

También se puede utilizar el "rest operator" en conjunción con destructuring para extraer los elementos remanentes, los elementos que queden, después de tomar los primeros identificados. Una aclaración: este tipo de operador con tres puntos, se parece al "spread operator" pero su función es diferente.

```
const [x, ...y] = ['a', 'b', 'c']; // x='a'; y=['b', 'c']
```