

DESTRUCTURING

“

Nos permite **extraer datos** de **arrays** y **objetos literales** de una manera más sencilla y fácil de implementar.



SIN USAR DESTRUCTURING

Para extraer datos de un **array**, es necesario crear una variable y asignarle un elemento del array usando el *operador de índice*.

{}

```
let colores = ['Rojo', 'Azul', 'Amarillo'];  
let azul = colores[1];
```

Para extraer datos de un **objeto**, es necesario que crear una variable y asignarle una *propiedad específica* de ese objeto.

{}

```
let auto = {marca: 'Ford', anio: 1998};  
let marcaAuto = auto.marca;
```

DESESTRUCTURANDO ARRAYS

Para desestructurar un **array**, declaramos una variable (podemos usar *var*, *let* o *const*), y entre **corchetes**, escribimos el nombre que queremos. Podemos declarar más de una variable, separando cada una con una coma **,**.

Luego igualamos esa estructura al array del cual queremos extraer los datos.

```
let colores = ['Rojo', 'Azul', 'Amarillo'];  
let [rojo, azul, amarillo] = colores;
```

CÓMO FUNCIONA

Partiendo de un **array** previamente definido, se transfiere cada dato a las variables que definimos nosotros.

Javascript le asignará a cada variable el dato extraído de la estructura que elijamos, **respetando el orden original**.

```
let array = ['Rojo', 'Azul', 'Amarillo'];
```

```
let [color1, color2, color3] = array;
```



CÓMO FUNCIONA

Si queremos saltar un valor, podemos dejar vacío el nombre de la variable que correspondería con esa posición.

```
let array = ['Rojo', 'Azul', 'Amarillo'];
```

```
let [color1, , color2] = array;
```



The diagram illustrates the assignment of array elements to variables. It shows two lines of code. The first line defines an array with three elements: 'Rojo', 'Azul', and 'Amarillo'. The second line defines three variables: color1, an empty space, and color2, and assigns them the values from the array. An orange arrow points from 'Rojo' to color1. A yellow arrow points from 'Azul' to the empty space. Another yellow arrow points from 'Amarillo' to color2. This visualizes how the second element of the array is skipped when assigning to the second variable.

DESESTRUCTURANDO OBJETOS

Para desestructurar un **objeto literal**, creamos una variable (*podemos usar var, let o const*), y entre **llaves**, declaramos el o los **nombres** de las **propiedades** que queremos extraer.

A esa estructura la igualamos al objeto del cual queremos extraer los datos.

```
let persona = {nombre: 'Laura', edad: 31, faltas: 3};  
let {nombre, edad} = persona;
```

{ }

CÓMO FUNCIONA

Partiendo de un objeto previamente definido, se transfiere cada propiedad o método a una o más variables que definamos.

Javascript le asignará a cada variable el valor de la propiedad que hayamos elegido.

```
let persona = {nombre: 'Laura', edad: 31, faltas: 3};
```

```
let {nombre, faltas} = persona;
```



CÓMO FUNCIONA

Es posible que en algún caso necesitemos cambiarle el nombre a la variable que estamos creando.

En ese caso a continuación de la propiedad que estamos extrayendo colocamos dos puntos `:` seguidos del nuevo nombre.

```
let persona = {nombre: 'Laura', edad: 31, faltas: 3};
```

```
let {nombre, faltas: totalFaltas} = persona;
```



La desestructuración **no modifica** el array u objeto literal de origen.

Su único objetivo es **copiar** los **valores** de una manera más práctica y rápida.

