

M1 - 6 - Objetos literales

Objetos literales

Los objetos nos permiten agrupar distintos valores dentro de una misma variable. Estos ‘valores’ son propiedades que van a ser definidas a través del método ‘clave:valor’, la clave será el nombre con el que vamos a identificar a esa propiedad, y el valor, como su propio nombre indica, es el valor que le damos a esa propiedad, ambos separados por ‘:’.

Crear un objeto literal

Para crear un objeto literal utilizamos las llaves ‘{ }’ y dentro de ellos almacenamos la información mediante el sistema clave:valor. Las claves su vez se separan por una coma ‘,’.

```
>> let persona = {  
    nombre: 'tintin',  
    edad: 18  
}
```

Los nombres con los que identificamos las propiedades **deben seguir las normas que utilizamos para nombrar una variable, camelCase..** Todo en minúsculas, el primer carácter no puede ser un número y si el nombre consta de más de una palabra lo que haremos será identificar el comienzo de la segunda palabra a través de una mayúscula, es decir, por cada palabra nueva una mayúscula. Por lo demás, en minúscula sería más correcto.

Aun así, existe una manera a través de la cual podemos introducir nombres de manera más literal y con la sintaxis que queramos. Para ello debemos introducir el nombre entre comillas, como si de un string se tratara.

```
>> let persona = {  
    'nombre de pila' : 'tintin'  
}  
! SyntaxError: missing : after property id [Saber más]  
>> let persona = {  
    'nombre de pila' : 'tintin'  
}
```

Recuperar claves

Para acceder a los valores de las claves de un objeto, utilizamos la nomenclatura **nombreObjeto.nombrePropiedad**. Veamos un ejemplo de cómo podemos acceder a las propiedades 'nombre' y 'edad' de nuestro objeto 'persona'.

```
>> persona
<-- ▶ Object { nombre: "tintin", edad: 18 }
>> persona.nombre
<-- "tintin"
>> persona.edad
<-- 18
```

En el caso de haber nombrado las propiedades entre comillas, para poder nombrarla como queramos, no podemos utilizar la nomenclatura **nombreObjeto.'nombre Atributo'**, tendremos que utilizar el formato **nombreObjeto['nombre atributo']**

```
>> let persona = {
    'nombre de pila' : 'tintin'
}
<-- undefined
>> persona.'nombre de pila'
! SyntaxError: missing name after . operator [Saber más]
>> persona['nombre de pila']
<-- "tintin"
```

Modificar claves

Para modificar/cambiar el valor de un atributo de un objeto, utilizamos la misma sintaxis que para mostrarlo `nombreObjeto.nombrePropiedad` y, a continuación indicamos el símbolo de asignación “`=`” seguido del nuevo valor que queremos asignarle.

```
>> let persona = { nombre: 'tintin', edad: 18};  
← undefined  
>> persona.edad  
← 18  
>> persona.edad = 19  
← 19
```

Añadir claves

Podemos añadir un atributo asignando un valor a un atributo de objeto inexistente: `persona.profesion = 'periodista'`; creará el atributo ‘profesion’ dentro del objeto persona y le asignará el valor ‘periodista’.

```
← ▼ {...}  
  |   edad: 19  
  |   nombre: "tintin"  
  |   ► <prototype>: Object { ... }  
>> persona.profesion = 'periodista';  
← "periodista"  
>> persona  
← ► Object { nombre: "tintin", edad: 19, profesion: "periodista" }
```

- Hay que tener cuidado ya que si queremos cambiar el valor de una variable y escribimos el atributo erróneamente, crearemos un nuevo atributo en vez de modificar el valor. Por ejemplo, `persona.edd = 18;` haría que persona tuviera un atributo llamado ‘edad’ con valor 19 y otro llamado ‘edd’ con valor 18.

```
>> persona
< - { ... }
    |   edad: 19
    |   nombre: "tintin"
    |   ► <prototype>: Object { ... }
>> persona.edd = 18
< 18
>> persona
< ► Object { nombre: "tintin", edad: 19, edd: 18 }
```

Eliminar claves

Podemos eliminar un atributo con el operador delete: **delete persona.profesion;** Hará que la variable persona deje de tener el atributo profesion.

```
>> delete persona.profesion
< true
>> persona
< - { ... }
    |   edad: 19
    |   nombre: "tintin"
    |   ► <prototype>: Object { ... }
```

Objetos anidados

Podemos introducir un objeto como el valor de uno de los atributos de un objeto.

```
>> let persona = {
    nombre: 'Tintín',
    edad: 18,
    mascota: {
        nombre: 'Milú',
        animal: 'perro'
    }
};
```

Para acceder al objeto 'mascota', haríamos como con el resto de propiedades : **persona.mascota**, y si quisiéramos acceder al nombre por ejemplo : **persona.mascota.nombre**.

```
>> > let persona = {  
    nombre: 'Tintín',  
    edad: 18,  
    mascota: {  
        nombre: 'Milú',...  
< undefined  
>> persona.mascota  
< > Object { nombre: "Milú", animal: "perro" }
```

Introducir valores a través de variables

Podemos definir el valor de una propiedad a través de variables. En el ejemplo siguiente tenemos un objeto perro con las propiedades 'nombre' y 'animal' y queremos introducir este objeto como una propiedad de nuestro objeto 'persona', para ello simplemente tenemos que utilizar el nombre de la variable, en este caso perro, para asignarle dicho objeto a una de las propiedades.

```
>> perro  
< > Object { nombre: "Milú", animal: "perro" }  
>> let persona = {  
    nombre: 'Tintín',  
    edad: 18,  
    mascota: perro  
}
```

También podemos asignarle a una propiedad un array como valor.

```
>> let comidaFavorita = ['hueso', 'chuleta', 'pollo'];  
< undefined  
>> let mascota = {  
    nombre: 'milú',  
    animal: 'perro',  
    comida: comidaFavorita  
};  
< undefined  
>> mascota.comida[1]  
< "chuleta"
```