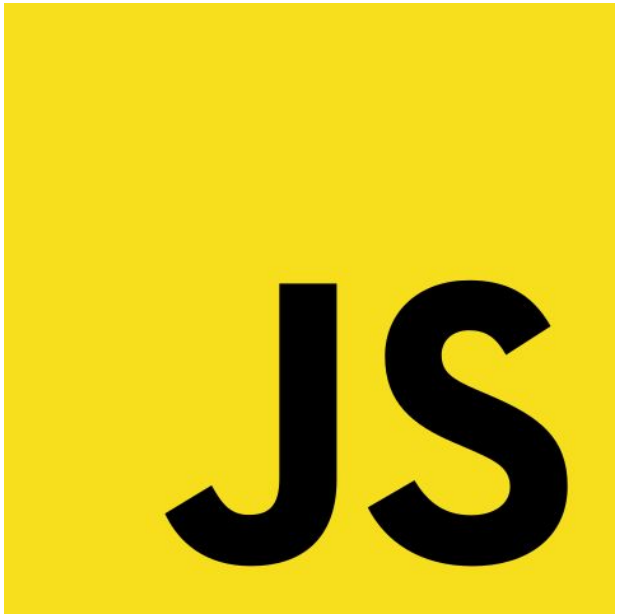
Objetos en Javascript



Un objeto es una colección de datos relacionados denominadas propiedades.

```
var perro = {
    nombre: 'Tyson',
    raza: 'Caniche Toy',
    edad: 10,

    ladrar: function () {
        console.log('Woof Woof!');
    }
}

console.log( perro.nombre );

// → Tyson
```

Edwin Rozo Gómez - edwin.rozo1@misena.edu.co

Objetos en Javascript

Los objetos al igual que todo lo que manejamos en javascript manejan su propia sintaxis. como se mostrará a continuación:

```
const product = {
  code: '001',
  detail: 'T-Shirt',
  price: '50USD',
  avalaibleColours: ['Blue','Black','Yellow','Red','Green'],
  stock: {
    adidas: 50,
    tierraSta: 80
  }
}
```

Nótese que dentro de sus propiedades podemos guardar distintos tipos de datos, incluyendo arreglos u otros objetos.

Edwin Rozo Gómez - edwin.rozo1@misena.edu.co

Agregar y eliminar elementos de un objeto en Javascript

Puedo agregar elementos al objeto de manera manual, pero también probablemente desee agregarlos desde la ejecución. De igual modo tenemos un método para la eliminación de propiedades en el objeto.

```
1  // Eliminar uno de los datos del objeto
2  delete product.code;
```

```
1  // Agregar elementos al objeto
2  product.image = 'Imagen.jpg';
```

Edwin Rozo Gómez -
edwin.rozo1@misena.edu.co

Destructuring de objetos en Javascript

Normalmente cuando deseo un elemento del objeto, sería de la siguiente forma:

objeto.propiedad;

Pero no siempre lo tendremos que hacer de esta manera ya que puedo de algún modo usar cada valor de manera independiente con **destructuring**

```
12    // Forma clásica
13    let response = `Su producto deseado es: ${product.detail} y cuesta ${product.price}`;
14    // Ayudándose de destructuring
15    let { detail, price } = product;
16    let response2 = `Su producto deseado es: ${detail} y cuesta ${price}`;
17
18    console.log(response);
19    console.log(response2);
20
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL bash

```
usuario@edwin-rozo:~/Documents/Rozo_Dev/Javascript/Basic/Javascript-Basic-2021/5.Objetos$ node Objetos.js
Su producto deseado es: T-shirt y cuesta 50USD
Su producto deseado es: T-shirt y cuesta 50USD
```

Edwin Rozo Gómez - edwin.rozo1@misena.edu.co

Destructuring de objetos en Javascript

Normalmente cuando deseo un elemento del objeto, sería de la siguiente forma:

objeto.propiedad;

Pero no siempre lo tendremos que hacer de esta manera ya que puedo de algún modo usar cada valor de manera independiente con **destructuring**

```
12    // Forma clásica
13    let response = `Su producto deseado es: ${product.detail} y cuesta ${product.price}`;
14    // Ayudándose de destructuring
15    let { detail, price } = product;
16    let response2 = `Su producto deseado es: ${detail} y cuesta ${price}`;
17
18    console.log(response);
19    console.log(response2);
20
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL bash

```
usuario@edwin-rozo:~/Documents/Rozo_Dev/Javascript/Basic/Javascript-Basic-2021/5.Objetos$ node Objetos.js
Su producto deseado es: T-shirt y cuesta 50USD
Su producto deseado es: T-shirt y cuesta 50USD
```

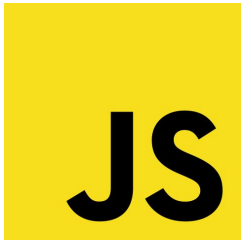
Edwin Rozo Gómez - edwin.rozo1@misena.edu.co

las propiedades de un objeto declarado const si se pueden modificar.

```
const user = {
  name: 'Edwin',
  age: 34,
  avalaible: true
}

console.log(user);

user.name = 'Gustavo';
//Detalles que no gustan mucho en js, es la posibilidad de cambiar de tipo de dato dentro de la mism variable.
user.age = 35;
user.avalaible = false;

console.log(user);

```

```
{ name: 'Edwin', age: 34, avalaible: true }
{ name: 'Gustavo', age: 35, avalaible: false }
```

Edwin Rozo Gómez -
edwin.rozo1@misena.edu.co



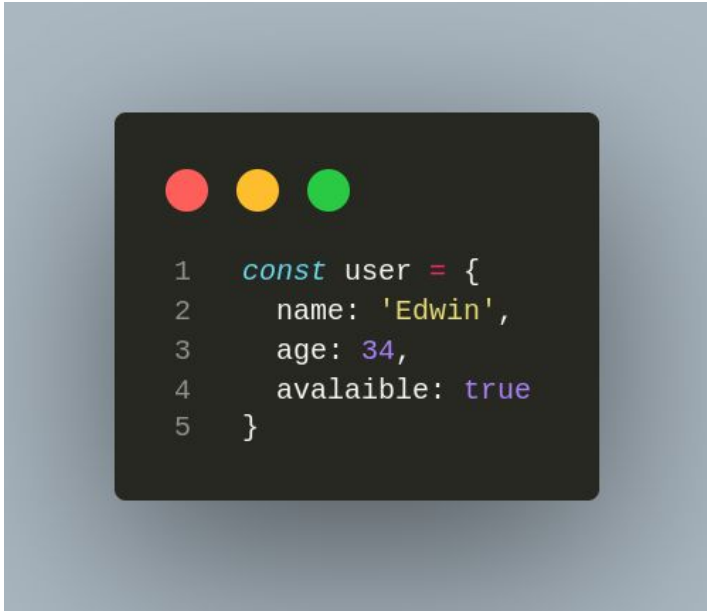
Métodos para objetos

“use strict” -	Habilitar el modo estricto, me permite usar ciertos métodos para objetos.
Object.freeze(myObject)	Congelar el objeto para que no pueda ser modificado
Object.isFrozen(myObject)	Comprueba si el objeto está congelado o no.
Object.seal(myObject)	Modificar llaves del objeto, pero no eliminarlas
Object.isSealed(myObject)	comprobar si el objeto está sellado.

Edwin Rozo Gómez -
edwin.rozo1@misena.edu.co

Unir 2 objetos

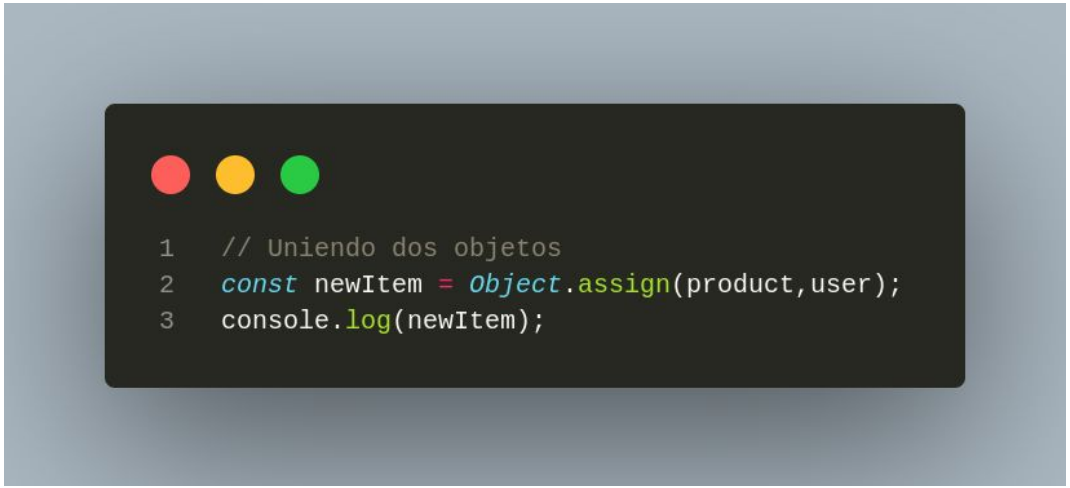
Algo que usaremos en algunos casos es unir dos objetos.
Ejemplo: objeto1 es **product** y objeto2 es **user**



Edwin Rozo Gómez -
edwin.rozo1@misena.edu.co

Unir 2 objetos

Algo que usaremos en algunos casos es unir dos objetos.
Ejemplo: objeto1 es **product** y objeto2 es **user**
con la sintaxis **Object.assign(product,user)** uiremos los dos objetos.



Edwin Rozo Gómez - edwin.rozo1@misena.edu.co

Unir 2 objetos

Como resultado:


```
{
  code: 1,
  detail: 'T-shirt',
  price: '50USD',
  avalaibleColours: [ 'Blue', 'Black', 'White', 'Yellow', 'Red', 'Green'
],
  stock: { adidas: 50, nike: 80 },
  name: 'Edwin',
  age: 34,
  avalaible: true
}
```

Edwin Rozo Gómez - edwin.rozo1@misena.edu.co

Unir 2 objetos

Podemos también añadir los dos objetos, utilizando spread operation**(...)**

```
1  let uniendoObjetos = {...product,...user};
2  console.log(uniendoObjetos);
```

Edwin Rozo Gómez - edwin.rozo1@misena.edu.co

Usando this en los objetos

La propiedad **this** tiene como finalidad dar contexto a determinado valor dentro del objeto. Tal como se ve en el ejemplo:

```
const product = {
  nombre: 'camisa',
  code: '001',
  price: 10,
  resumen: function () {
    console.log(`El producto ${this.nombre} tiene un precio de ${this.price}`);
  }
}
```

Edwin Rozo Gómez - edwin.rozo1@misena.edu.co