DESARROLLO WEB EN ENTORNO CLIENTE

OBJETOS DEFINIDOS POR EL USUARIO

4. PROGRAMACIÓN CON OBJETOS DEFINIDOS POR EL USUARIO

Objetos:

En Javascript, un objeto es una colección de propiedades que a su vez pueden ser datos o métodos.

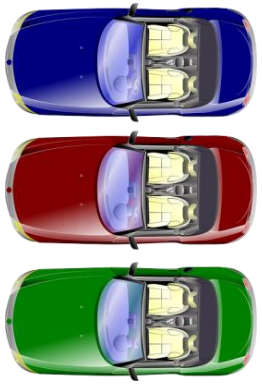
Constructor: es una función especial para crear un objeto. Empieza en mayúscula:

```
function Coche()
{ //Propiedades y métodos}
```

Creación de un objeto:

```
let unCoche = new Coche();
```

Coche()



```
let cocheazul = new Coche();
let cocherojo = new Coche();
let cocheverde = new Coche();
```

4. PROGRAMACIÓN CON OBJETOS DEFINIDOS POR EL USUARIO

Definir propiedades del objeto:

se crean en el constructor precedidas por la palabra reservada "this".

"Constructor" sin parámetros:

```
function Coche()
{
  this.marca ="”; //Vacío
  this.modelo = “”; //Vacío
  this.combustible  =“Diesel”; //Inicializado
  this.cantidad=0;  //Inicializado
}
```

"Constructor" con parámetros:

```
function  Coche(marca,  modelo, combustible,  cantidad)
{
  this.marca =marca;
  this.modelo = modelo;
  this.combustible  =  combustible;
  this.cantidad=cantidad;
  //Cada propiedad  toma  los valores recibidos  por parámetro
}
```

Crear un objeto vacío (sin propiedades):

```
let cocheVacio  = new  Coche();
```

Cambiar valores en las propiedades:

```
cocheVacio.marca  = “Seat”;
cocheVacio.modelo = “Ibiza”;
cocheVacio.combustible  = “Diesel”;
cocheVacio.cantidad   = 40;
```

Crear un objeto inicializado (con propiedades):

```
let miCoche = new  Coche(“Seat”,  “Ibiza”,  “Diesel”,  40);
```

Acceder a las propiedades:

```
document.write(“ Mi coche  es un
“+miCoche.marca+”  “+miCoche.modelo);
```

4. PROGRAMACIÓN CON OBJETOS DEFINIDOS POR EL USUARIO

Definir métodos del objeto:

Permiten acceder y modificar las propiedades de los objetos. Empiezan en minúscula.

Método que modifica la cantidad de combustible

```
function rellenarDeposito(litros)
{
  this.cantidad = litros;
}
```

Referencia al método fuera del objeto

(¡No recomendado!)

```
function Coche(marca, modelo, combustible, cantidad) {
  //Propiedades
  this.marca =marca;

  this.modelo = modelo;

  this.combustible = combustible;

  //Métodos
  this.rellenardeposito = rellenardeposito;
}
```

Referencia al método dentro del objeto

```
function Coche(marca, modelo, combustible, cantidad) {
  //Propiedades
  this.marca = marca;
  this.modelo= modelo;
  this.combustible = combustible;

  //Métodos
  this.rellenardeposito = function (litros) {
      this.cantidad = litros;
  }
}
```

4. PROGRAMACIÓN CON OBJETOS DEFINIDOS POR EL USUARIO

Objetos literales:

Un literal es un valor fijo. Está formado por parejas de tipo nombre:valor.

Ejemplo:

```
coche {marca:"Ibiza", modelo:"Seat", combustible:"diesel", cantidad:40};
```

Ejemplo equivalente:

```
let coche = new Object();
coche.marca="Ibiza";
coche.modelo="Seat";  coche.combustible="diesel";
coche.cantidad=40;
```

Acceso:

```
coche.marca;
coche["marca"];
```

4. PROGRAMACIÓN CON OBJETOS DEFINIDOS POR EL USUARIO

EJERCICIO: u4e4_objetos:

Necesitamos almacenar en un programa todos los discos de música que tenemos en casa. Ahora que sabemos crear nuestros propios objetos es el mejor modo de guardar esta información.

- *Crea un objeto "disco" que almacene la siguiente información:*
  - *Nombre del disco.*
  - *Grupo de música o cantante.*
  - *Año de publicación.*
  - *Tipo de música (podrá ser "rock", "pop", "punk" o "indie");*
  - *Localización: almacenará un número de estantería.*
  - *Prestado: almacenará un valor booleano. Por defecto será false.*
- *Además tendrá los siguientes métodos:*
  - *Un "constructor" sin parámetros (las 4 primeras propiedades serán cadenas vacías, la localización será 0 por defecto y prestado estará a false).*
  - *Un método que permita incluir las cinco primeras propiedades; la propiedad prestado seguirá a false.*
  - *Un método que permitirá cambiar el número de estantería en la localización.*
  - *Un método que permitirá cambiar la propiedad Prestado.*
  - *Un método que muestre toda la información de un disco.*
- *Guarda todo el código en un archivo llamado disco.js*