



PRÁCTICA 1: CLASES Y OBJETO JAVASCRIPT

1. INTRODUCCIÓN

Desde la norma de **ECMAScript5** Javascript puede considerarse un lenguaje de programación orientado a objetos, pero con algunas peculiaridades.

No posee todas las características de otros lenguajes orientados a objetos como Java o C++, pero si es capaz de manejar y crear objetos. Además permite la herencia, pero no la sobrecarga de funciones.

2. OBJETOS JAVASCRIPT

Los objetos son una colección de propiedades. Para construir objetos podemos hacerlo de dos maneras,

- **Objetos declarativos o literales:** podemos crear objetos sin necesidad de un constructor o instanciar una clase, para esto solo declaramos el objeto y sus propiedades.

```
const camilo = {
  nombre: 'Camilo',
  edad: 22,
  sexo: 'masculino',
  pasatiempos: ['patinar', 'bailar'],
  hablar: function(){
    return `hola soy ${this.nombre}, y tengo ${this.edad} años`;
  }
}
```

```
console.log(camilo);
```

- **Objetos contruidos:** JavaScript es un lenguaje libre de clases, pero tenemos el keyword new, el cual nos permite crear un nuevo objeto, de esta manera podemos utilizar una función que cumpla el rol del constructor.

```
function Persona(nombre, edad, sexo, pasatiempos) {
  this.nombre = nombre;
  this.edad = edad;
  this.sexo = sexo;
  this.pasatiempos = pasatiempos;
  this.hablar = function() {
    return `hola soy ${this.nombre}, y tengo ${this.edad} años`;
  };
}
const camilo = new Persona('camilo', 22, 'masculino', ['patinar', 'bailar']);
console.log(camilo)
```

3. CLASES EN JAVASCRIPT

Las clases de javascript, introducidas en **ECMAScript 2015**, son una mejora sintáctica sobre la herencia basada en prototipos de JavaScript. La sintaxis de las clases no introduce un nuevo modelo de herencia orientada a objetos en JavaScript. Las clases de JavaScript proveen una sintaxis mucho más clara y simple para crear objetos y lidiar con la herencia

4. CONSTRUCTORES EN JAVASCRIPT

El siguiente código muestra un constructor para la clase coche:

```
var coche(modelo, color, kms, combustible) {
    this.modelo= modelo;
    this.color=color;
    this.kms=kms;
    this.combustible=combustible;
}

};

var elmio=new coche("Mercedes", "Negro", 120000,"diésel");
var eltuyo= new coche("BMW", "blanco", 210000, "gasolina");
```

El método **constructor** es un método especial para crear e inicializar un objeto creado con una clase. Solo puede haber un método especial con el nombre "constructor" en una clase. Si esta contiene mas de una ocurrencia del método constructor, se arrojará un *Error*

Un **constructor** puede usar la *palabra reservada* super para llamar al **constructor** de una *superclase*

```
class Square extends Polygon {
    constructor(length) {
        // Aquí, llama al constructor de la clase padre con sus longitudes
        // contemplando la anchura y la altura del Polígono
        super(length, length);
        // Nota: En las clases derivadas, super() se debe llamar primero
        // Se puede utilizar "this". Dejando esto causará un error de
        //referencia.
        this.name = 'Square';
    }
    get area() {
        return this.height * this.width;
    }
    set area(value) {
        this.area = value;
    }
}
```

Si no especifica un método constructor, se utiliza un constructor predeterminado. Para las clases base, el constructor por defecto es **super**



5. EJEMPLO DE CLASE: CLASE VEHICULO

```
'use strict'
//Clase Vehiculo
class Vehiculo {
  constructor(modelo, marca, precio, km) {
    this.modelo = modelo;
    this.marca = marca;
    this.precio = precio;
    this.km=km;
  }

  getModelo(){
    return this.modelo;
  }

  setModelo(modelo){
    this.modelo=modelo;
  }

  getMarca(){
    return this.marca;
  }

  setMarca(marca){
    this.marca=value;
  }

  getPrecio(){
    return this.precio;
  }

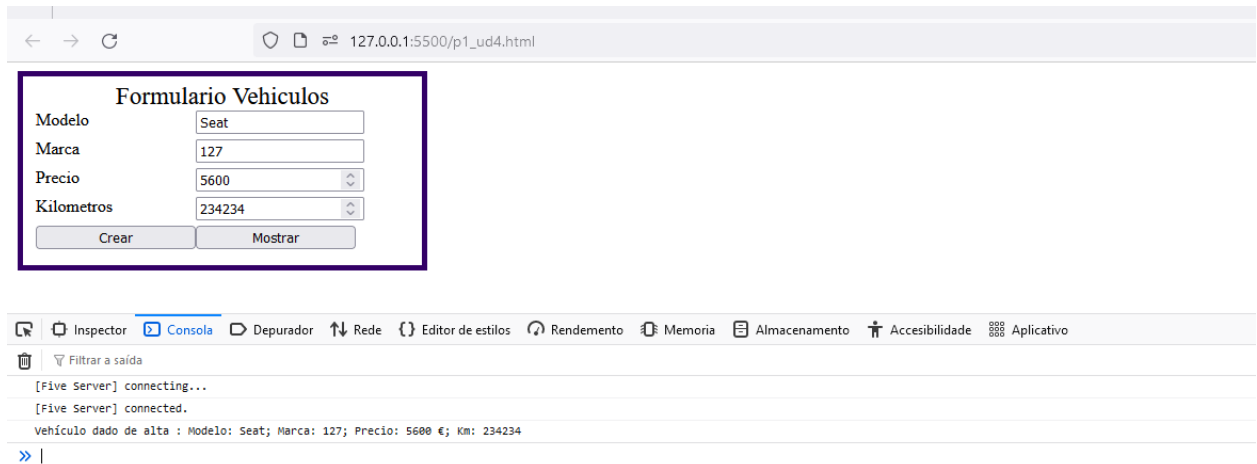
  setPrecio(value){
    this.precio=value;
  }
  //metodos
  mostrarDatos() {
    return "El vehiculo " + this.modelo;
  }
}

//Declarar un arrayDeCoches

let arrayDeCoches=[new Vehiculo("Trans","Seat",12.000,120.000),
new Vehiculo("Passat","Volswagen",12.000,120.000),
new Vehiculo("C5","Citroen",12.000,120.000),
new Vehiculo("Micra","Hyundai",12.000,120.000)]
```

Secuencia y desarrollo:

- Ejercicio 1:** Utilizando la clase Vehículo, vamos a crear una aplicación en JavaScript para gestionar un aparcamiento de coches.



The screenshot shows a web browser window with the address bar displaying '127.0.0.1:5500/p1_ud4.html'. The main content area contains a form titled 'Formulario Vehiculos' with the following fields and buttons:

- Modelo:** Text input with 'Seat' entered.
- Marca:** Text input with '127' entered.
- Precio:** Text input with '5600' entered.
- Kilometros:** Text input with '234234' entered.
- Buttons:** 'Crear' and 'Mostrar' buttons at the bottom.

Below the form, the browser's developer console is open, showing the following output:








```
[Five Server] connecting...
[Five Server] connected.
Vehículo dado de alta : Modelo: Seat; Marca: 127; Precio: 5600 €; Km: 234234
>> |
```

Funcionamiento:

- Pedir los datos de cada vehículo con un formulario
- Al pulsar un botón **Guardar** se almacena los vehículos en un **ArrayList** de vehículos
- Añadir un botón **Mostrar** que visualice todos los coches aparcados en el **aparcamiento**. Para ello debes crear un método **imprimirVehiculo** que muestre los datos utilizando los métodos de creación de nodos de la estructura DOM.

Formulario Vehiculos

Modelo	<input type="text" value="Seat"/>
Marca	<input type="text" value="127"/>
Precio	<input type="text" value="5600"/>
Kilometros	<input type="text" value="234234"/>

-  Vehículo: 1; Modelo: Seat; Marca: 127; Precio: 5600 €; Km: 234234
-  Vehículo: 2; Modelo: Ibiza; Marca: Seat; Precio: 8500 €; Km: 125000
-  Vehículo: 3; Modelo: 506; Marca: Preugeot; Precio: 10000 €; Km: 50000
-  Vehículo: 4; Modelo: A3; Marca: Audi; Precio: 15000 €; Km: 100000
-  Vehículo: 5; Modelo: Corsa; Marca: Opel; Precio: 10000 €; Km: 50000
-  Vehículo: 6; Modelo: Laguna; Marca: Renault; Precio: 13000 €; Km: 75000
-  Vehículo: 7; Modelo: X5; Marca: BMW; Precio: 27000 €; Km: 50000