



Unidad 5

```
/*  
Unidad 5: Programación Orientada a Objetos (POO) y Objetos predefinidos en JavaScript
```

```
1. Todo en JavaScript es un objeto:
```

```
*/
```

```
/* Objeto literal con propiedades y métodos */
```

```
let persona = {  
  nombre: "Juan",  
  edad: 25,  
  saludar: function() {  
    console.log("Hola, " + this.nombre);  
  }  
};  
persona.saludar(); // "Hola, Juan"
```

```
/*
```

```
2. JSON (JavaScript Object Notation)
```

```
*/
```

```
// Ejemplo JSON como texto
```

```
let jsonStr = '{"nombre":"Ana","edad":30}';
```

```
// Convertir texto JSON a objeto JS
```

```
let obj = JSON.parse(jsonStr);
```

```
console.log(obj.nombre); // "Ana"
```

```
// Convertir objeto JS a texto JSON
```

```
let nuevoStr = JSON.stringify(obj);
```

```
console.log(nuevoStr); // '{"nombre":"Ana","edad":30}'
```

```

/*
3. Acceso a propiedades y métodos
*/

console.log(persona.nombre); // Acceso con punto: "Juan"
console.log(persona["edad"]); // Acceso con corchetes: 25
persona.saludar(); // Invocación método

```

```

/*
4. Operador instanceof
*/

console.log(persona instanceof Object); // true

```

```

/*
5. Constructores y creación de objetos
*/

function Telefono(marca, modelo) {
  this.marca = marca;
  this.modelo = modelo;
  this.descripcion = function() {
    return this.marca + " " + this.modelo;
  }
}

let tel1 = new Telefono("Samsung", "A30");
console.log(tel1.descripcion()); // "Samsung A30"

```

```

/*
6. Herencia y polimorfismo con clases ES6
*/

class Miembro {
  constructor(nombre, estado) {
    this.nombre = nombre;
    this.estado = estado;
  }
}

```

```

    }
    cobrar() {
        console.log(this.nombre + " cobra su salario base.");
    }
}

class Profesor extends Miembro {
    constructor(nombre, estado, numAlumnos) {
        super(nombre, estado);
        this.numAlumnos = numAlumnos;
    }
    cobrar() {
        console.log(
            this.nombre + " cobra salario de profesor por " + this.numAlumnos + " a
            lumnos."
        );
    }
}

class Alumno extends Miembro {
    constructor(nombre, estado, asignaturas) {
        super(nombre, estado);
        this.asignaturas = asignaturas;
    }
    cobrar() {
        console.log(this.nombre + " recibe beca de alumno.");
    }
}

let prof = new Profesor("Laura", "activo", 40);
let alum = new Alumno("Carlos", "activo", 5);

prof.cobrar(); // "Laura cobra salario de profesor por 40 alumnos."
alum.cobrar(); // "Carlos recibe beca de alumno."

/*
7. Recorrer propiedades de un objeto
*/

```

```

for (let prop in persona) {
  console.log(prop + ": " + persona[prop]);
}

/*
8. Eliminar propiedades
*/

delete persona.edad;
console.log(persona.edad); // undefined

/*
9. Prototipos
*/

// Definir prototipo
function Animal(nombre) {
  this.nombre = nombre;
}
Animal.prototype.hablar = function () {
  console.log(this.nombre + " hace un ruido.");
};

let perro = new Animal("Toby");
perro.hablar(); // "Toby hace un ruido."

/*
10. Objetos predefinidos

a) String
*/

let texto = "Hola Mundo";
console.log(texto.toUpperCase()); // "HOLA MUNDO"
console.log(texto.split(" ")); // ["Hola", "Mundo"]

/*

```

b) Date

*/

```
let ahora = new Date();
console.log(ahora.toLocaleDateString()); // Fecha local
let cumpleaños = new Date(1990, 4, 15);
console.log(cumpleaños.getFullYear()); // 1990
```

/*

c) Math

*/

```
console.log(Math.PI); // 3.141592653589793
console.log(Math.round(3.6)); // 4
console.log(Math.random()); // Número aleatorio entre 0 y 1
```

/*

d) Boolean

*/

```
let verdadero = true; // primitivo
let falsoObjeto = new Boolean(false);
console.log(verdadero); // true
console.log(falsoObjeto.valueOf()); // false
```

/*

e) Expresiones regulares (RegExp)

*/

```
// Validar si texto contiene solo dígitos
let regex = /^\d+$/;
console.log(regex.test("12345")); // true
console.log(regex.test("12a45")); // false
```

```
// Extraer palabras con exec
```

```
let texto2 = "Hola mundo";
let palabras = /\w+/g;
let match;
```

```
while ((match = palabras.exec(texto2)) !== null) {  
  console.log(match[0]);  
}  
  
/*  
Ejemplo expresión regular para validar email:  
*/  
let emailRegex = /^[^s@]+@[^s@]+\.[^s@]{2,}$;/  
console.log(emailRegex.test("correo@dominio.com")); // true  
console.log(emailRegex.test("correo@dominio")); // false
```