Evidencias Propiedades del objeto

```html
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Calidad del Curso - JavaScript.info</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #ffe0e6; /* Rosa claro para el fondo */
            margin: 0;
            padding: 20px;
        }
        h1 {
            color: #ff69b4; /* Rosa más oscuro para el título principal */
            text-align: center;
            margin-bottom: 30px;
        }
        section {
            background-color: #fff;
            border-radius: 8px;
            box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
            margin-bottom: 20px;
            padding: 20px;
        }
        h2 {
            color: #ff1493; /* Rosa brillante para los subtítulos de sección */
            border-bottom: 2px solid #ffc0cb; /* Línea inferior rosa claro */
            padding-bottom: 10px;
            margin-top: 0;
            margin-bottom: 15px;
        }
        p {
            color: #333;
            line-height: 1.6;
        }
        .code-output {
```

```html
    </style>
</head>
<body>
    <h1>Calidad del Curso</h1>

    <!-- Sección para el tema 7.1 -->
    <section id="tema-7-1">
        <h2>7.1 Banderas y descriptores de propiedad</h2>
        <p>Aquí exploraremos cómo las propiedades de un objeto tienen atributos especiales (ba
        <div class="code-output" id="output-7-1">
            <!-- Los resultados de JavaScript se inyectarán aquí -->
        </div>
    </section>

    <!-- Sección para el tema 7.2 -->
    <section id="tema-7-2">
        <h2>7.2 Captadores y configuradores de propiedades</h2>
        <p>En esta sección, veremos cómo los "getters" (captadores) y "setters" (configurador
        <div class="code-output" id="output-7-2">
            <!-- Los resultados de JavaScript se inyectarán aquí -->
        </div>
    </section>

    <script src="main.js"></script>
</body>
</html>
```

```
    <p class="console-output">Mira la consola (F12) para más detalles de los descrip
`;


// TEMA: 7.2 Captadores y configuradores de propiedades (Getters y Setters) - 27/10/
// Este bloque explora cómo usar métodos "getter" (captador) y "setter" (configurado
// para definir propiedades de objeto que se comportan como funciones cuando se leen
// Esto permite añadir lógica personalizada al acceso de propiedades.

// 1. Objeto con getter y setter para una propiedad 'fullName'
const person = {
    firstName: "linda",
    lastName: "sofia",

    get fullName() {
        console.log("Getter de fullName ejecutado.");
        return `${this.firstName} ${this.lastName}`;
    },

    set fullName(value) {
        console.log("Setter de fullName ejecutado con valor:", value);
        [this.firstName, this.lastName] = value.split(' ');
    }
};

// 2. Acceder a la propiedad fullName (llama al getter)
console.log("Nombre completo de la persona (usando getter):", person.fullName);
// Aprendizaje: Acceder a 'person.fullName' invoca automáticamente la función 'get f
```

```
// 2. Acceder a la propiedad fullName (llama al getter)
console.log("Nombre completo de la persona (usando getter):", person.fullName);
// Aprendizaje: Acceder a 'person.fullName' invoca automáticamente la función 'get fullName

// 3. Asignar un valor a la propiedad fullName (llama al setter)
person.fullName = "Carlos Ruiz";
console.log("Nuevo nombre completo (después de usar setter):", person.fullName);
console.log("Nuevo firstName:", person.firstName);
console.log("Nuevo lastName:", person.lastName);
// Aprendizaje: Asignar un valor a 'person.fullName' invoca automáticamente la función 'set

// 4. Ejemplo práctico: Propiedad 'age' con validación en el setter
const userWithAge = {
    _age: 0, // Usamos un prefijo _ para indicar que es una propiedad interna

    set age(value) {
        if (value < 0) {
            console.error("La edad no puede ser negativa.");
        } else {
            this._age = value;
            console.log("Edad asignada:", value);
        }
    },

    get age() {
        return this._age;
    }
};

// Asignar una edad válida
userWithAge.age = 15;
console.log("Edad del usuario (válida):", userWithAge.age);

// Intentar asignar una edad inválida
userWithAge.age = -5;
console.log("Edad del usuario (después de intentar inválida):", userWithAge.age);
```

Calidad del Curso

7.1 Banderas y descriptores de propiedad

Aquí exploraremos cómo las propiedades de un objeto tienen atributos especiales (banderas) que controlan su comportamiento, y cómo podemos ver y modificar estos descriptores.

Objeto 'user' con 'name':

```
{
  "name": "sofia"
}
```

Descriptor de 'user.name' por defecto:

```
{
  "value": "sofia",
  "writable": true,
  "enumerable": true,
  "configurable": true
}
```

Valor de 'user.name' después de setear 'writable: false' y intentar cambiarlo: sofia