Practica 04-05c Object.freeze y Object.seal

Nombre		Curso	
Apellidos		Fecha	

Inmutabilidad con Object.freeze y Object.seal aunque no ofrecen inmutabilidad profunda como Immutable.js, sí ayudan a limitar la mutación de objetos en JavaScript.

1. Object.freeze() — Congelación total (inmutabilidad superficial)

Object.freeze(obj) hace que **un objeto NO pueda cambiarse en absoluto**:

- No se pueden **agregar** propiedades
- No se pueden **eliminar** propiedades
- No se pueden **modificar** propiedades existentes
- No se puede **cambiar su prototipo**
- No se puede **reasignar** valores a sus claves

Pero OJO: solo congela *el objeto del primer nivel*.
Si tiene objetos dentro → esos siguen siendo mutables.

✔️ Ejemplo:

```
const user = Object.freeze({
  name: "Ana",
  settings: {
    theme: "dark"
  }
});

user.name = "Pedro";     // ❌ no cambia
user.age = 30;           // ❌ no se agrega
delete user.name;        // ❌ no se elimina

user.settings.theme = "light"; // ✅ esto sí cambia (NO es inmutable profundo)
```

Conclusión: Inmutabilidad **superficial**.

Si quieres inmovilizar todo, deberías congelar cada nivel manualmente o usar una función recursiva.

2. Object.seal() — Sellado (el objeto puede cambiar, pero no crecer)

Object.seal(obj) permite **modificar valores**, pero **prohíbe agregar o borrar propiedades**:

- No se pueden **agregar** propiedades
- No se pueden **eliminar** propiedades
- Sí se pueden **modificar valores** de claves existentes
- Tampoco se puede cambiar el prototipo

✔️ Ejemplo:

```
const user = Object.seal({
  name: "Ana",
  age: 20
});

user.age = 21;   // ✔️ permitido
user.name = "Luis"; // ✔️ permitido

user.city = "Madrid"; // ❌ no se puede agregar
delete user.name;     // ❌ no se puede borrar
```

Conclusión: No es inmutable. Solo evita modificaciones estructurales.

Comparación rápida

Característica	freeze	seal
Modificar valores	❌ No	✔️ Sí
Agregar propiedades	❌ No	❌ No
Eliminar propiedades	❌ No	❌ No
Inmutabilidad	❌	❌ No

Característica	freeze	seal
profunda	No	
Cambiar prototipo	❌ No	❌ No

¿Sirven para inmutabilidad real?

No del todo.

- freeze da **inmutabilidad superficial**
- seal solo impide crecer, pero **no evita cambiar valores**

Para inmutabilidad real en estructuras complejas se prefiere:

- **Immutable.js**
- **Immer.js**
- **Estructuras estructuralmente compartidas**
- **Clonado profundo** (menos eficiente)

Ejemplo de inmutabilidad profunda con freeze

Si quieres congelar todo el árbol:

```
function deepFreeze(obj) {
  Object.freeze(obj);

  Object.values(obj)
    .filter(value => typeof value === "object" && value !== null)
    .forEach(value => deepFreeze(value));

  return obj;
}

const user = deepFreeze({
  name: "Ana",
  settings: { theme: "dark" }
});

user.settings.theme = "light"; // ❌ ahora sí falla
```

Resumen final

- Object.freeze() → inmutabilidad superficial; no permite cambiar NADA del objeto pero sí de objetos internos.
- Object.seal() → permite cambiar valores, pero no agregar ni borrar propiedades.
- Ninguna de las dos ofrece inmutabilidad profunda como las librerías dedicadas.

EJEMPLO 1 — Uso real de Object.freeze()

```
const user = Object.freeze({
  name: "Ana",
  age: 25,
  settings: {
    theme: "light"
  }
});

// Intentos de modificar (NO funcionan)
user.age = 30;          // ❌ ignorado
user.city = "Madrid";   // ❌ ignorado
delete user.name;       // ❌ ignorado

console.log(user.age);   // 25

// PERO objetos internos sí cambian
user.settings.theme = "dark"; // ✔️ permitido

console.log(user.settings.theme); // "dark"
```

Punto clave: Object.freeze() congela SOLO el primer nivel.

EJEMPLO 2 — Uso real de Object.seal()

```
const product = Object.seal({
  id: 1,
  name: "Laptop",
  price: 1000
});

// Puedes modificar valores existentes
product.price = 900;      // ✔️ permitido
product.name = "Laptop Pro"; // ✔️ permitido

// Pero NO puedes agregar ni eliminar propiedades
```


```
product.stock = 20;       // ❌ ignorado
delete product.id;        // ❌ ignorado

console.log(product);
```

Punto clave: seal() es más flexible pero NO crea inmutabilidad.

PRÁCTICA

NIVEL 1 — Básico

Ejercicio 1: Datos de un estudiante

Crea un objeto:

{ name: "Luis", grade: 7, approved: false }

1. Congélalo con Object.freeze.
2. Intenta:
    - cambiar la nota
    - agregar una propiedad age
    - eliminar la propiedad approved

👉 Comprueba que nada cambia.

Ejercicio 2: Sellar configuración

Crea un objeto:

{ volume: 50, brightness: 80 }

1. Súbelo una propiedad contrast = 100 ANTES de sellarlo.
2. Luego usa Object.seal().
3. Intenta cambiar los valores existentes.
4. Intenta agregar otro campo o borrar uno.

👉 Observa qué funciona y qué no.

NIVEL 2 — Inmutabilidad profunda

Ejercicio 3: Congelación profunda

Crea una función deepFreeze(obj) que:

- congele el objeto con Object.freeze()

- si detecta valores que son objetos, también los congele recursivamente

Después:

1. Aplica deepFreeze() a este objeto:

```
{
  user: {
    name: "Ana",
    preferences: {
      theme: "dark"
    }
  }
}
```

2. Intenta cambiar user.name y preferences.theme.

👉 Ambos intentos deben fallar.

Ejercicio 4: Diferencias freeze vs seal

Crea dos objetos iguales:

```
const car1 = { brand: "Ford", year: 2010 };
const car2 = { brand: "Ford", year: 2010 };
```

Haz lo siguiente:

1. Aplica freeze a car1.
2. Aplica seal a car2.
3. Luego intenta:
    - cambiar year en ambos
    - agregar propiedad color
    - eliminar propiedad brand

👉 Anota qué pasa en cada caso.

NIVEL 3 — Casos reales

Ejercicio 5: Estado de una app

Imagina un estado Redux:

```
const state = {
  settings: {
    notifications: true,
```


```
    theme: "light"
  },
  user: {
    name: "Carlos",
    loggedIn: true
  }
};
```

1. Congélalo superficialmente con freeze().
2. Cambia loggedIn y observa que NO cambia.
3. Cambia theme y observa que SÍ cambia.
4. Explica por qué ocurre esto.

👉 (pista: mutación profunda vs superficial)

Ejercicio 6: Sellar un modelo

Crea un objeto:

```
const book = { id: 1, title: "1984", author: "Orwell" };
```

1. Sella el objeto.
2. Implementa una función updateBook(obj, newTitle) que:
    - intente cambiar el título
    - intente agregar propiedad pages
    - intente eliminar author
3. Comprueba qué cambios se aplican.