

# Code Smell 1: Bloaters - Large Class

Ubicación: src/models/Vehiculo.java

## Antes de la refactorización:

```
package models;

public class Vehiculo {

    private float velocidadActual;
    private boolean vehiculoEncendido;
    private Llanta llanta;
    private Motor motor;

    public Vehiculo(Motor motor, Llanta llanta) {
        this.velocidadActual = 0;
        this.vehiculoEncendido = false;
        this.llanta = llanta;
        this.motor = motor;
    }

    public void encender() {
        if (vehiculoEncendido) {
            throw new VehiculoYaEncendidoException();
        }
        vehiculoEncendido = true;
    }

    public void apagar() throws Exception {
        if (!vehiculoEncendido) {
            throw new VehiculoYaApagadoException();
        }
        if (velocidadActual > 60) {
            vehiculoEncendido = false;
            velocidadActual = 0;
            throw new ApagadoVelocidadAltaException();
        }
        vehiculoEncendido = false;
    }

    public float acelerar(float incremento) throws Exception {
        if (!vehiculoEncendido) {
            throw new VehiculoApagadoException();
        }
        velocidadActual += incremento;
        return velocidadActual;
    }

    public float frenar(float decremento) throws Exception {
        if (!vehiculoEncendido) throw new VehiculoApagadoException();
        if (velocidadActual == 0) throw new FrenadoEnDetenidoException();
        if (decremento > velocidadActual) throw new FrenadoIntensidadInvalidaException();
        velocidadActual -= decremento;
        return velocidadActual;
    }
}
```

## Después de la refactorización:

```
// Se divide en sistemas especializados

public class SistemaEncendido { ... }
```

```

public class SistemaAceleracion { ... }

public class SistemaFrenos { ... }

public class Vehiculo {

    private float velocidadActual;
    private boolean vehiculoEncendido;
    private Llanta llanta;
    private Motor motor;

    private SistemaEncendido sistemaEncendido;
    private SistemaAceleracion sistemaAceleracion;
    private SistemaFrenos sistemaFrenos;

    public Vehiculo(Motor motor, Llanta llanta) {
        this.velocidadActual = 0;
        this.vehiculoEncendido = false;
        this.llanta = llanta;
        this.motor = motor;

        this.sistemaEncendido = new SistemaEncendido();
        this.sistemaAceleracion = new SistemaAceleracion();
        this.sistemaFrenos = new SistemaFrenos();
    }
}

```

## Breve explicación:

Se identificó el code smell Bloaters - Large Class en la clase Vehiculo, ya que concentraba múltiples responsabilidades: encendido, apagado, aceleración y frenado. Se aplicó la refactorización Extract Class, dividiendo el comportamiento en sistemas especializados, mejorando la cohesión y reduciendo el acoplamiento.

---

## Code Smell 2: Bloaters - Long Method

**Ubicación:** src/models/Vehiculo.java - método frenar()

### Antes de la refactorización:

```

public float frenar(float decremento) throws Exception {

    if (!vehiculoEncendido) throw new VehiculoApagadoException();

    if (velocidadActual == 0)
        throw new FrenadoEnDetenidoException();

    if (decremento > velocidadActual)
        throw new FrenadoIntensidadInvalidaException();

    velocidadActual -= decremento;

    if (velocidadActual < 0)
        velocidadActual = 0;
}

```

```
        return velocidadActual;
    }
}
```

## Después de la refactorización:

```
public float frenar(float decremento) throws Exception {
    validarEncendido();
    validarEstado();
    validarIntensidad(decremento);
    velocidadActual = calcularNuevaVelocidad(decremento);
    return velocidadActual;
}

private void validarEncendido() throws VehiculoApagadoException { ... }
private void validarEstado() throws FrenadoEnDetenidoException { ... }
private void validarIntensidad(float decremento)
    throws FrenadoIntensidadInvalidaException { ... }
private float calcularNuevaVelocidad(float decremento) { ... }
```

## Breve explicación:

Se identificó el code smell Bloaters - Long Method, ya que el método frenar contenía múltiples validaciones y lógica en un solo bloque extenso. Se aplicó Extract Method para dividir el método en funciones más pequeñas, mejorando legibilidad, mantenibilidad y claridad estructural.

---

## Code Smell 3: Dispensables - Duplicated Code

**Ubicación:** src/views/VentanaAcelerar.java, VentanaFrenar.java

## Antes de la refactorización:

```
// Código repetido en varias ventanas (VentanaAcelerar, VentanaFrenar)

boton.addActionListener(e -> {
    try {
        vehiculo.acelerar(10);
    } catch (Exception ex) {
        JOptionPane.showMessageDialog(null, ex.getMessage());
    }
});
```

## Después de la refactorización:

```
public void configurarBoton(JButton boton, Runnable accion) {
```

```
        boton.addActionListener(e -> {  
            try {  
                accion.run();  
            } catch (Exception ex) {  
                JOptionPane.showMessageDialog(null, ex.getMessage());  
            }  
        });  
    }  
}
```

## Breve explicación:

Se identificó el code smell Dispensables - Duplicated Code, debido a la repetición de estructuras similares en varias clases de vista. Se aplicó Extract Method centralizando la lógica repetida en un método reutilizable, reduciendo duplicidad y facilitando mantenimiento futuro.

---