**Ejemplo: Clase "Vehiculo" y sus subclases**

Imaginemos que tenemos una clase base llamada Vehiculo y dos subclases: Coche y Motocicleta. Queremos implementar un método en la clase base que determine el tipo de vehículo y realice acciones específicas basadas en ese tipo.

**1. Clase Base: Vehiculo**

Java

public class Vehiculo {

private String marca;

private String modelo;

public Vehiculo(String marca, String modelo) {

this.marca = marca;

this.modelo = modelo;

}

public String getMarca() {

return marca;

}

public String getModelo() {

return modelo;

}

// Método que usa if-else

public void mostrarTipo() {

if (this instanceof Coche) {

System.out.println("Este vehículo es un coche.");

} else if (this instanceof Motocicleta) {

System.out.println("Este vehículo es una motocicleta.");

} else {

System.out.println("Este vehículo es de tipo desconocido.");

}

}

}

**2. Subclase: Coche**

Java

public class Coche extends Vehiculo {

private int numeroPuertas;

public Coche(String marca, String modelo, int numeroPuertas) {

super(marca, modelo);

this.numeroPuertas = numeroPuertas;

}

public int getNumeroPuertas() {

return numeroPuertas;

}

// Sobreescritura del método (opcional)

@Override

public void mostrarTipo() {

System.out.println("Este vehículo es un coche con " + numeroPuertas + " puertas.");

}

}

**3. Subclase: Motocicleta**

Java

public class Motocicleta extends Vehiculo {

private int cilindrada;

public Motocicleta(String marca, String modelo, int cilindrada) {

super(marca, modelo);

this.cilindrada = cilindrada;

}

public int getCilindrada() {

return cilindrada;

}

// Sobreescritura del método (opcional)

@Override

public void mostrarTipo() {

System.out.println("Este vehículo es una motocicleta con cilindrada de " + cilindrada + "cc.");

}

}

**4. Clase Principal: Main**

Java

public class Main {

public static void main(String[] args) {

Vehiculo vehiculo1 = new Coche("Toyota", "Corolla", 4);

Vehiculo vehiculo2 = new Motocicleta("Honda", "CBR", 600);

Vehiculo vehiculo3 = new Vehiculo("Genérico", "Desconocido");

vehiculo1.mostrarTipo();

vehiculo2.mostrarTipo();

vehiculo3.mostrarTipo();

// Si sobreescribimos el método en las subclases, se usará el método de la subclase

Coche coche1 = new Coche("Ford", "Mustang", 2);

coche1.mostrarTipo();

}

}

**Explicación:**

* **Herencia:**
  + Coche y Motocicleta heredan de Vehiculo.
* **instanceof:**
  + En el método mostrarTipo() de la clase Vehiculo, utilizamos instanceof para verificar el tipo de objeto.
  + if (this instanceof Coche) verifica si el objeto actual es una instancia de la clase Coche.
* **if-else:**
  + Utilizamos if-else para ejecutar diferentes bloques de código según el tipo de vehículo.
* **Polimorfismo (Sobreescritura de Métodos):**
  + Opcionalmente, podemos sobreescribir el método mostrarTipo() en las subclases para proporcionar una implementación específica para cada tipo de vehículo. Esto se demuestra en los ejemplos de las clases Coche y Motocicleta.
* **Constructores, Getters y Setters:**
  + Se utilizan constructores para inicializar los objetos y getters para acceder a los atributos.

**Puntos clave para tus alumnos:**

* **Herencia:** Cómo las subclases heredan atributos y métodos de la clase base.
* **Polimorfismo:** Cómo un objeto puede tomar múltiples formas.
* **instanceof:** Cómo verificar el tipo de un objeto en tiempo de ejecución.
* **if-else:** Cómo tomar decisiones basadas en el tipo de objeto.
* **Encapsulamiento:** El uso de atributos privados y métodos públicos.