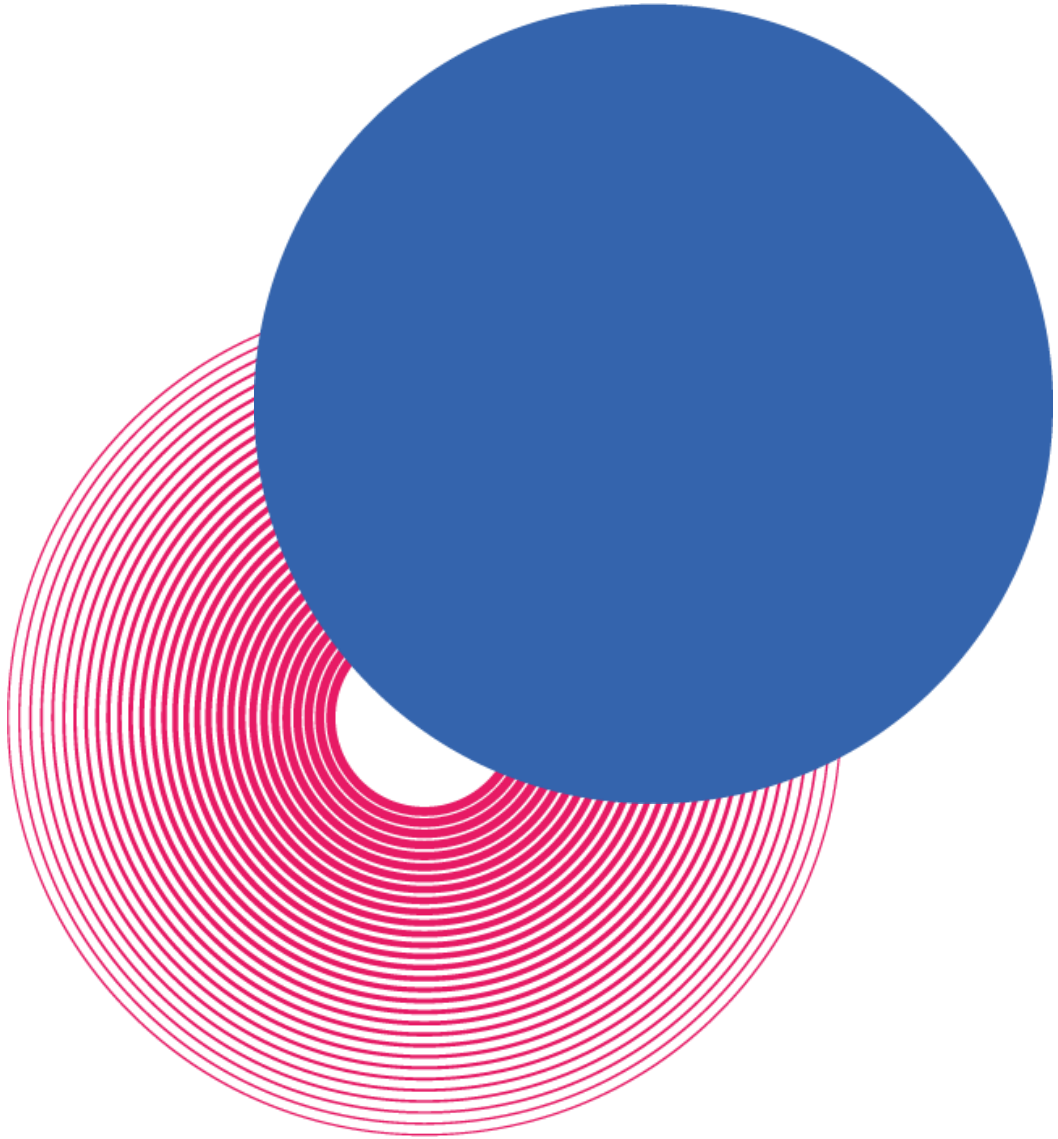
HERENCIA



Definición

La ***herencia** es uno de los mecanismos fundamentales de la programación orientada a objetos, por medio del cual una clase se construye a partir de otra.

Una de sus funciones más importantes es proveer el ***polimorfismo** (puro, sobreescritura, sobrecarga)**.**

**Mas adelante, a través del ejemplo de la clase Mascota se comprenden más fácilmente dichos conceptos.*

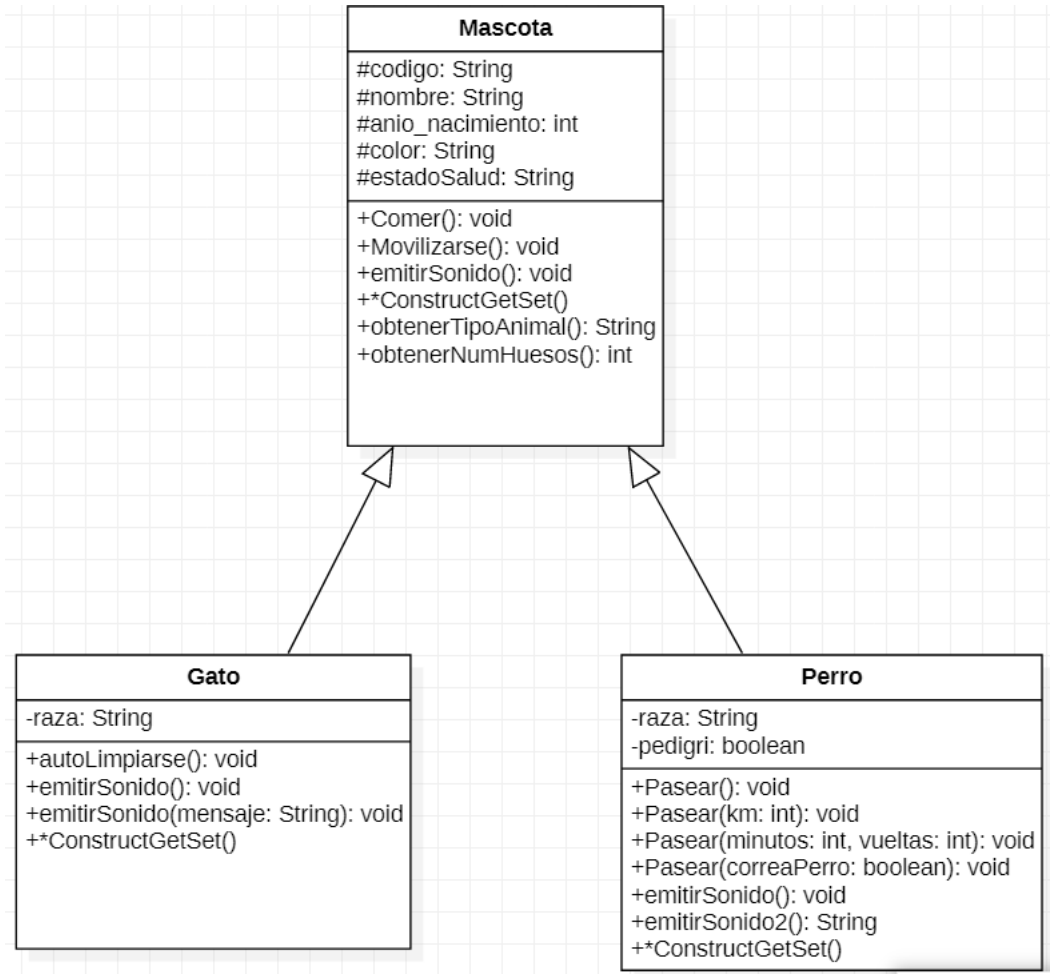
Definición

Relaciona las clases de manera jerárquica:

Una clase padre (superclase / clase base) con otras clases hijas (subclases / clase derivada).

En **Java** solo se admite herencia simple (una clase puede heredar solamente de una superclase), en cambio C++ si se acepta herencia múltiple (puede haber ambigüedades porque una clase puede heredar de dos o más superclases)

A nivel de código en POO, para especificar la superclase se realiza con la palabra **extends.**


Mascota
#codigo: String
#nombre: String
#anio_nacimiento: int
#color: String
#estadoSalud: String
+Comer(): void
+Movilizarse(): void
+emitirSonido(): void
+*ConstructGetSet()
+obtenerTipoAnimal(): String
+obtenerNumHuesos(): int
Gato
-raza: String
+autoLimpiarse(): void
+emitirSonido(): void
+emitirSonido(mensaje: String): void
+*ConstructGetSet()
Perro
-raza: String
-pedigri: boolean
+Pasear(): void
+Pasear(km: int): void
+Pasear(minutos: int, vueltas: int): void
+Pasear(correaPerro: boolean): void
+emitirSonido(): void
+emitirSonido2(): String
+*ConstructGetSet()

Cuando hablamos de programas de **Java**, también se usa la expresión “extiende”, pues Java utiliza la palabra clave **extends** para definir la relación de herencia.

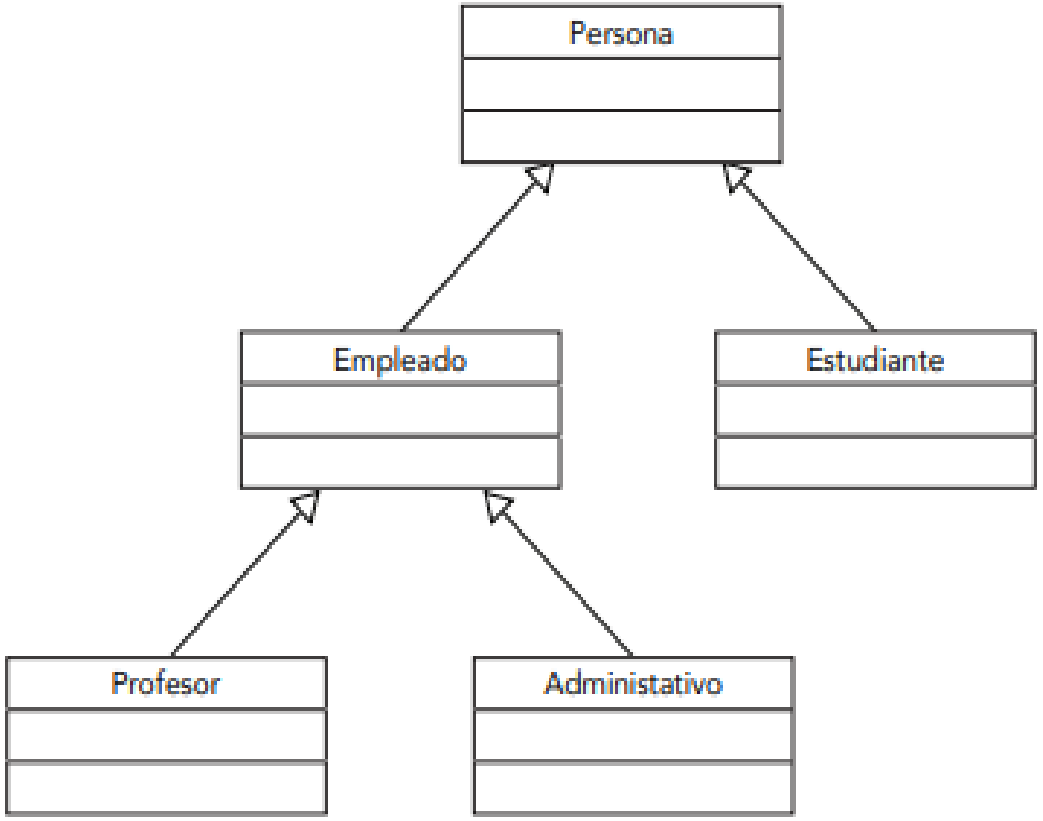
La flecha en el diagrama de clases (dibujada generalmente con la punta sin rellenar) representa la relación de herencia. Dicha flecha va dirigida desde la clase derivada (específica) hacia la clase base (general). Por tal razón, en UML la relación de herencia se conoce como **generalización**.

Algunas veces, la herencia también se denomina relación «es un».

Concepto de superclase y subclase

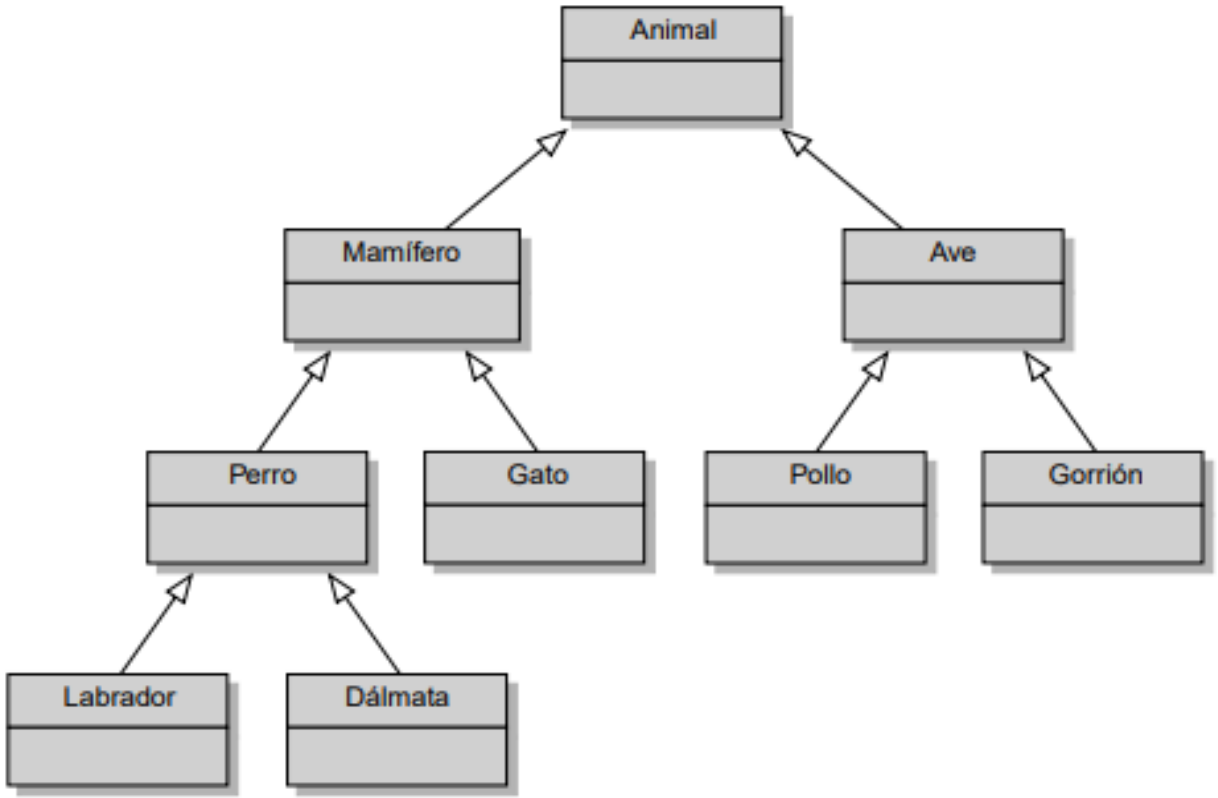
La generalización es una relación entre clases en las que hay una clase padre, llamada **superclase**, y una o más clases hijas especializadas, a las que se les denomina **subclases**.

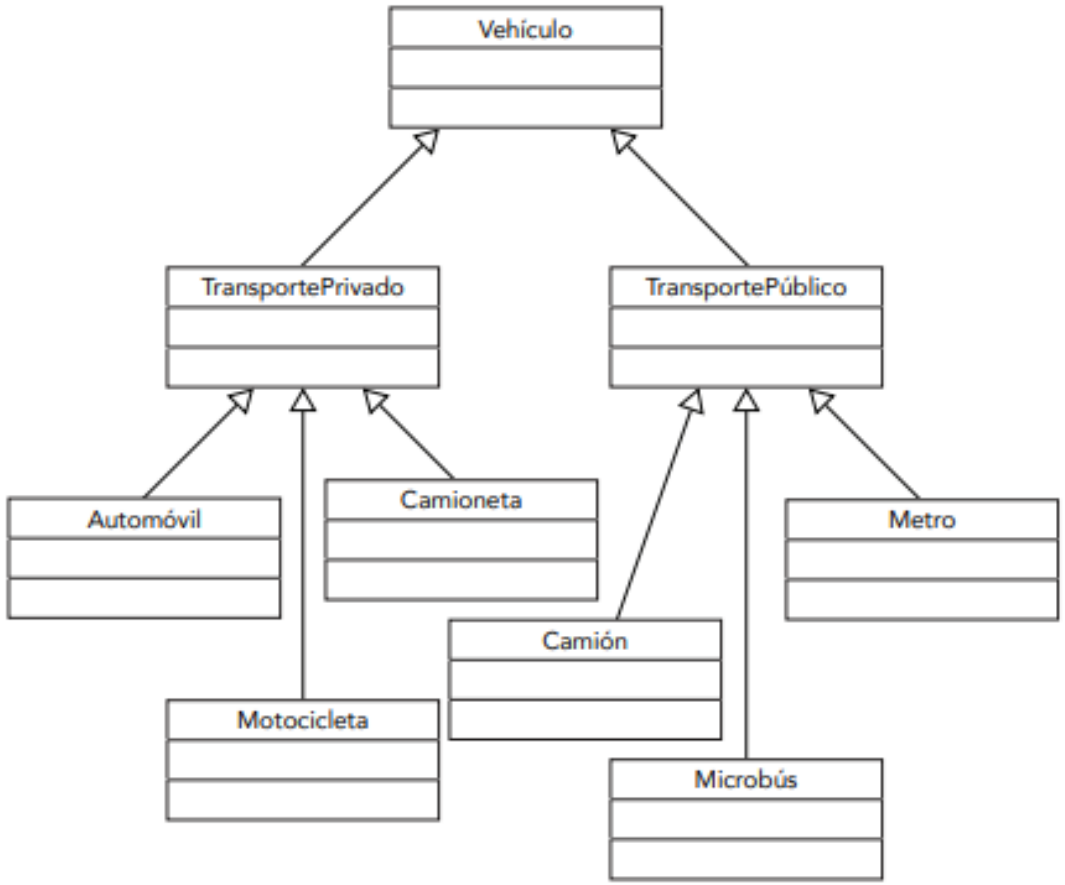
La herencia es el mecanismo mediante el cual se implementa la relación de generalización.


Persona
Empleado
Estudiante
Profesor
Administativo

Jerarquías de clase

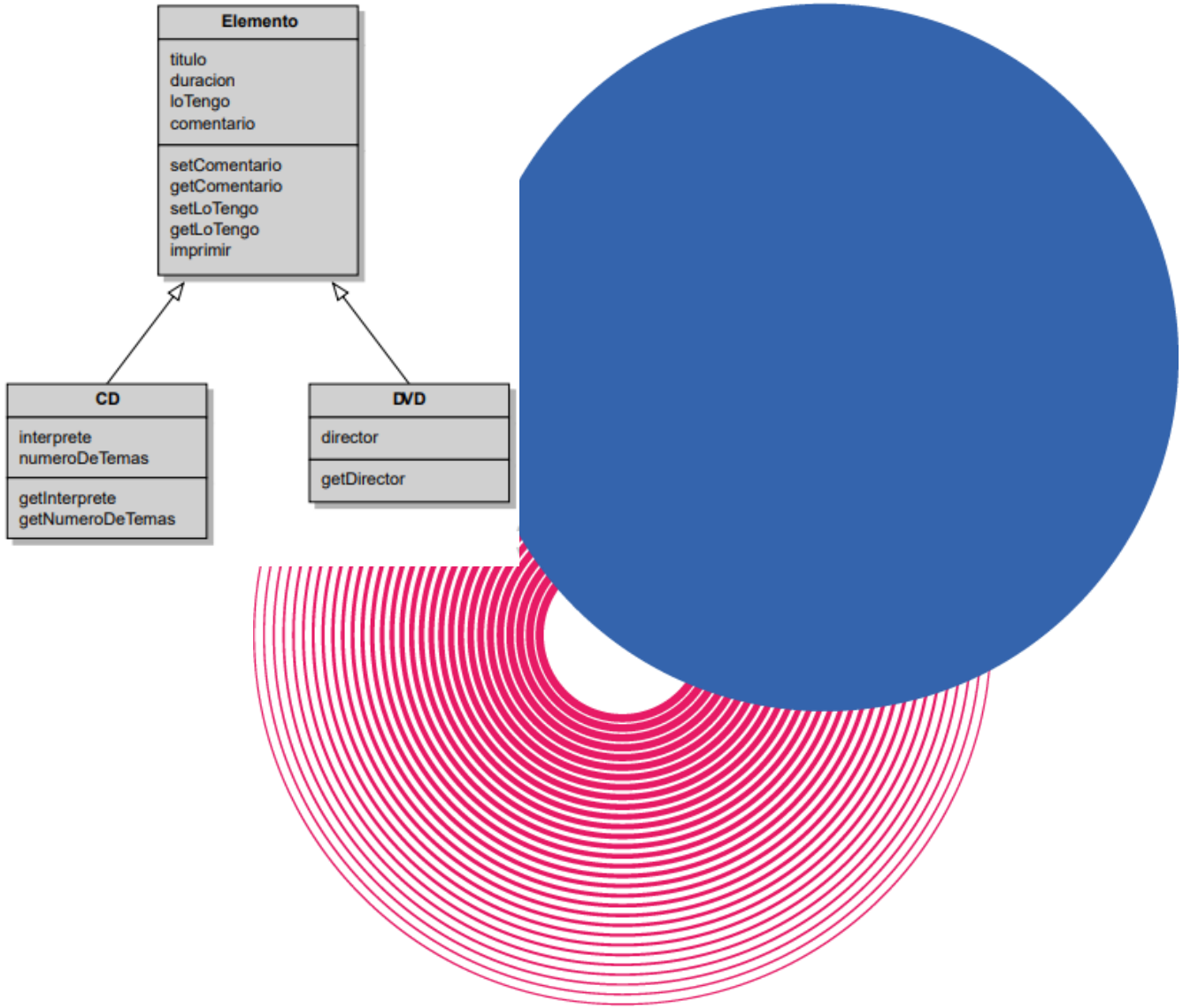
Se pueden heredar más de dos subclases a partir de la misma superclase y una subclase puede convertirse en la superclase de otras subclases ***(Java: HERENCIA SIMPLE)***


Animal
Mamífero
Ave
Perro
Gato
Pollo
Gorrión
Labrador
Dálmata


Vehículo
TransportePrivado
TransportePúblico
Automóvil
Camioneta
Metro
Camión
Motocicleta
Microbús

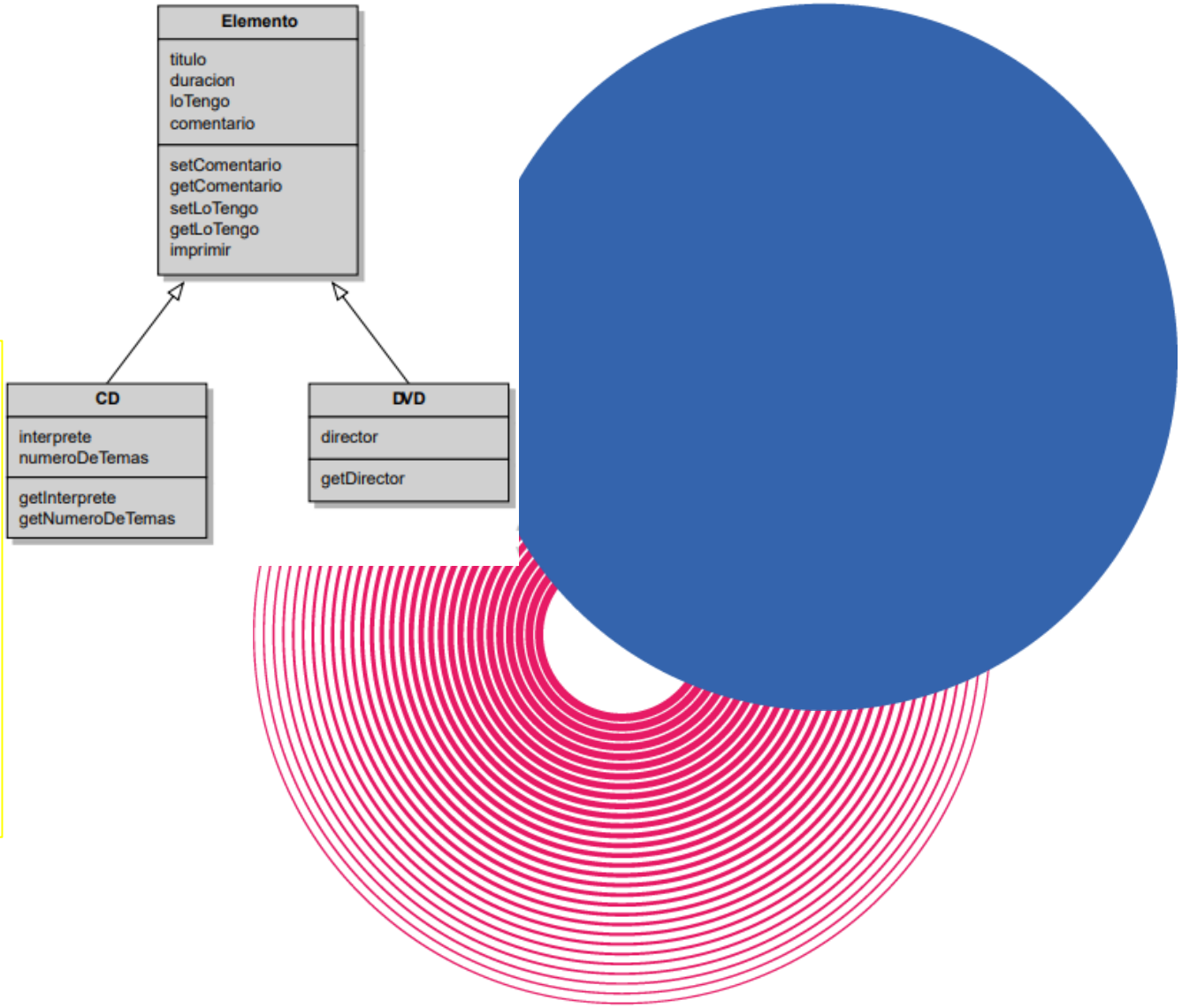
Java

```
public class Elemento
{
  private String titulo;
  private int duracion;
  private boolean lotengo;
  private String comentario;
  // se omitieron constructores y métodos
}
```

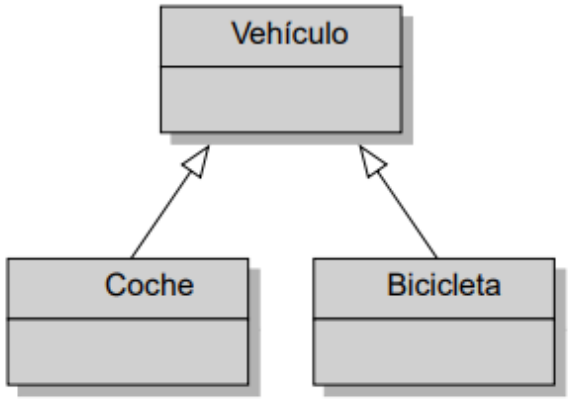


Java

```java
public class CD extends Elemento
{
  private String interprete;
  private int numeroDeTemas;
  /**
  * Constructor de objetos de la clase CD
  * @param elTitulo El título del CD.
  * @param elInterprete El intérprete del CD.
  * @param temas El número de temas del CD.
  * @param tiempo La duración del CD.
  */
  public CD(String elTitulo, String elInterprete, int temas, int tiempo)
  {
      super(elTitulo, tiempo);
      interprete = elInterprete;
      numeroDeTemas = temas;
  }
// Se omitieron métodos
}
```



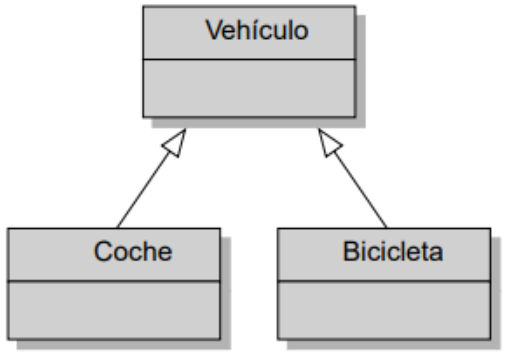
Subtipos y asignación



```
coche   miCoche  =  new  Coche();
```

Subtipos y asignación

```
Vehiculo v1 = new Vehiculo();
Vehiculo v2 = new Coche();
Vehiculo v3 = new Bicicleta();
```



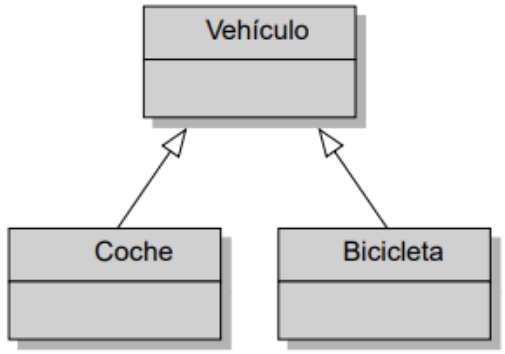
¿Por qué es útil la herencia?

1. Reutilización de código.

2. Creación de programas extensibles.

Subtipos y asignación

```
Coche a1 = new Vehiculo();      // ¡Es un error!
```



FIN