



Diagramas de Clases UML.

Karen juliana Castillo.

Gestor del conocimiento.

William Alexander Matallana Parra.

Universidad de Cundinamarca, extensión Chía.

Ingeniería de Sistemas y computación.

11 de febrero de 2026.



Contenido.

Diagramas de Clases UML.....	1
Programación orientada a objetos (POO)	3
Los cuatro pilares de POO.....	3
Diagrama de clases UML.	4
Encapsulamiento.....	4
Relaciones entre clases.	4
Ejemplo de diagrama de clases.....	7
Conclusiones.....	8
Referencias.	9



Programación orientada a objetos (POO)

Se basa en crear tipos nuevos de datos llamados objetos, los atributos son las propiedades que definen al objeto y los métodos las acciones que pueden realizar, para especificar que se quiere crear un tipo de dato nuevo se utiliza la clase que es donde se definen los atributos y métodos.

El método instanciar sirve para crear el objeto y darle los valores necesarios para inicializarlo, con ayuda del constructor.

Los cuatro pilares de POO.

Abstracción: En esta se añaden los atributos y métodos necesarios para cumplir el objetivo del programa.

Encapsulamiento: Protege los atributos y métodos de una clase, controlando que es accesible o no desde fuera, se declaran atributos y métodos como privados y permitiendo el acceso con métodos públicos como get (para acceder) y set (para modificar).

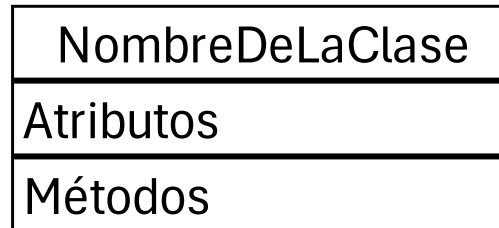
Herencia: Permite crear nuevas clases a partir de una clase existente llamada clase padre, reutilizando sus atributos y métodos, esta clase que hereda se llama clase hija o subclase.

Polimorfismo: Permite que un mismo método tenga diferentes comportamientos dependiendo de la que lo implemente, es decir, puede ejecutarse de diferentes formas dependiendo del objeto que lo esté usando.



Diagrama de clases UML.

Una clase se modela de esta forma.



En el primer rectángulo se escribe el nombre de la clase, la primera en mayúscula.

En el segundo agregan los atributos en forma de lista junto al tipo de dato que son, no se colocan valores a estos.

En el tercero se colocan los constructores y lo métodos de la clase.

Encapsulamiento.

Los símbolos para indicar el nivel de acceso son:

-: privado

+: publico

#: protegido

Relaciones entre clases.

Asociación: Es una conexión entre dos clases relacionadas de forma lógica.

Notación: Clase1 — Clase2

Ejemplo: Cliente — Compra



Asociación dirigida (unidireccional). Indica que una clase conoce a la otra, pero no viceversa.

Notación: Clase1 \longrightarrow Clase2

Asociación reflexiva. Una clase se relaciona consigo misma.

Notación: Empleado \longrightarrow Empleado

Ejemplo: Empleado supervisa empleado

Multiplicidad. Indica cuántas instancias de una clase se relacionan con otra.

Notación: Flota 1 ----- * Avión (una flota tiene muchos aviones)

Notación: Avión 1 ----- * Pasajero (un avión puede tener muchos pasajeros)

Agregación. Relación "tiene-un" donde las partes pueden existir independientemente del todo.

Notación: Todo $\diamond\text{-----}$ Parte(diamante vacío)

Ejemplo: 'Departamento $\diamond\text{-----}$ Profesor' (un profesor puede existir sin el departamento)

Composición. Relación "tiene-un" más fuerte donde las partes no pueden existir sin el todo.

Notación: Todo $\blacklozenge\text{-----}$ Parte(diamante relleno)

Ejemplo: Casa $\blacklozenge\text{-----}$ Habitación (una habitación no existe sin la casa)

Herencia (Generalización). Relación "es-un" donde una clase hija hereda de una clase padre.

Notación: ClaseHija $\text{-----}\triangleright$ ClasePadre(flecha triangular vacía)

Ejemplo: Perro $\text{-----}\triangleright$ Animal



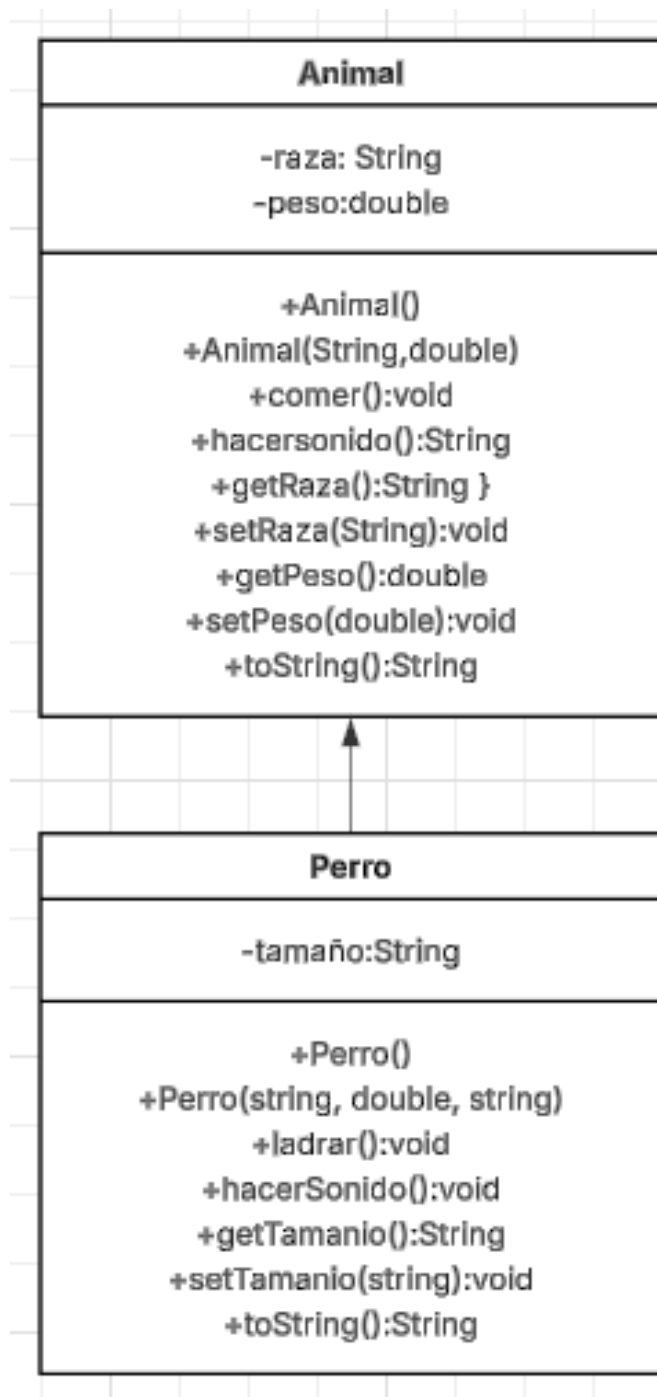
Realización (Implementación). Una clase implementa una interfaz.

Notación: Clase▷ Interfaz (línea punteada con flecha triangular vacía)

Ejemplo: ArrayList▷ List



Ejemplo de diagrama de clases.





Conclusiones.

Los diagramas de clases UML son muy importantes a la hora de realizar un proyecto de programación, ya que nos ayudan a plantear y observar todo el panorama para poder ejecutarlo de la mejor manera.

En la programación orientada a objetos (POO) es el pilar para poder llevar a cabo un buen código ya que la estructura organizacional es muy eficiente y permite una ejecución mas dinámica.



Referencias.

YouTube. (s.f). Programación Orientada a Objetos (POO): Abstracción, Encapsulamiento, Herencia, Polimorfismo [Video]. YouTube.

<https://youtu.be/SI7O81GMG2A>

YouTube. (s.f). DIAGRAMA de CLASES UML ¡Empezá por acá! [Video].

YouTube. <https://youtu.be/txxU2x5e3HM>

YouTube. (s.f). Relaciones entre Clases [Video]. YouTube.

<https://youtu.be/kpSP1W7T5VQ>