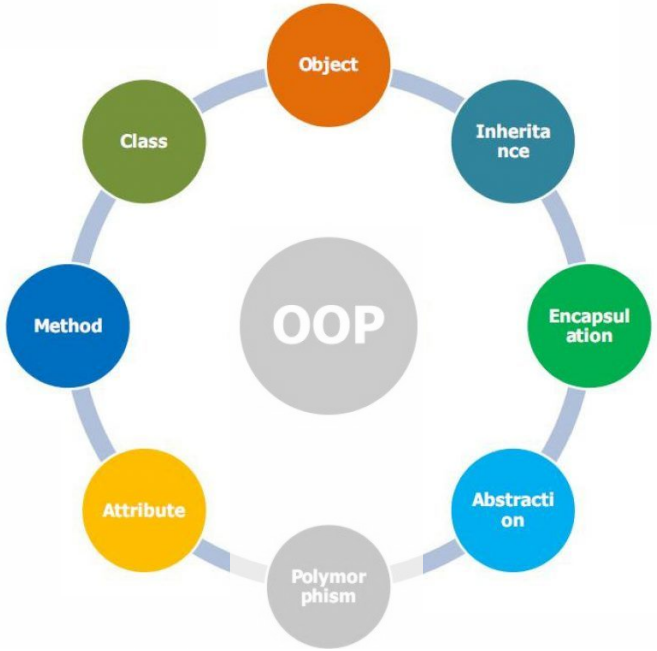
Herencia

Volvamos a nuestro diagrama y veamos ahora la relación de Herencia.



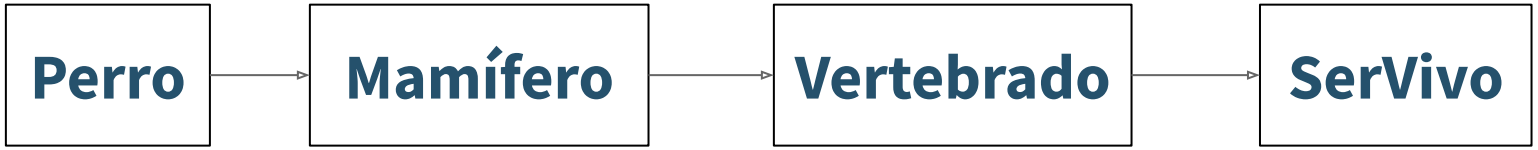
Herencia

Cuando hablamos de Herencia, estamos hablando de Jerarquías de Clases.

Un ejemplo de jerarquía: un **Perro**, es un **Mamífero**, que a la vez es un **Vertebrado**, que a la vez es un **SerVivo**.

Herencia

Perro **hereda** de Mamífero, que **hereda** de Vertebrado, que **hereda** de SerVivo.



Podemos decir, por ejemplo:

- Todo **Perro** es un **Vertebrado**
- Todo **Mamifero** es un **SerVivo**
- No todo **SerVivo** es **Mamífero** (Pato)
- No todo **Vertebrado** es **Perro** (Vaca), ni **Mamífero** (cocodrilo)

Herencia

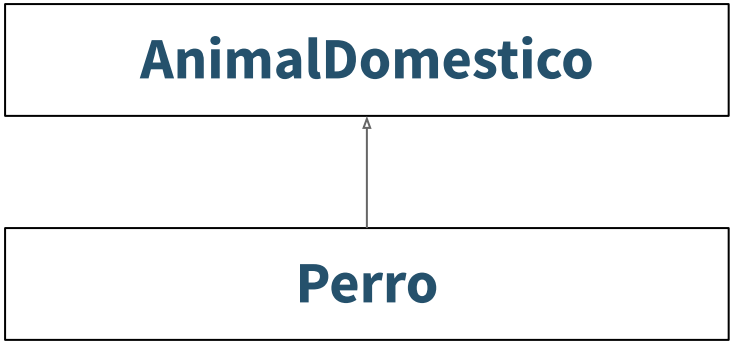
¿Qué es la herencia en OOP? → Jerarquías de clases.
A través de la Herencia, podremos crear nuevas clases a partir de otra preexistente (creo **Perro**, a través de **AnimalDomestico**).

Estas nuevas clases, en líneas generales, tendrán todos los atributos y métodos de su “padre”, y tendrán tambień otros más específicos, que serán su *razón de ser*.

Herencia

Así por ejemplo, **AnimalDomestico** tendrá como atributo **Duenio**, y como método **comer()**, ya que todos los animales domésticos tienen dueño y comen.
Entonces **Perro**, al heredar de **AnimalDomestico**, tendrá automáticamente el atributo **Duenio** y el método **comer()**.
Pero Perro además podrá tener el método **moverLaCola()**, que no tendría sentido en un **Canario** por ejemplo.

Herencia



El “padre” es más general.

El “hijo” es más específico.

El “hijo” es igual al padre, pero suma su especificidad.
Perro hace y tiene todo lo que hace y tiene **AnimalDomestico**, pero además puede **moverCola();**

UML - Herencia

Volvamos a UML. Veamos como diagramar las herencias.

UML - Herencia

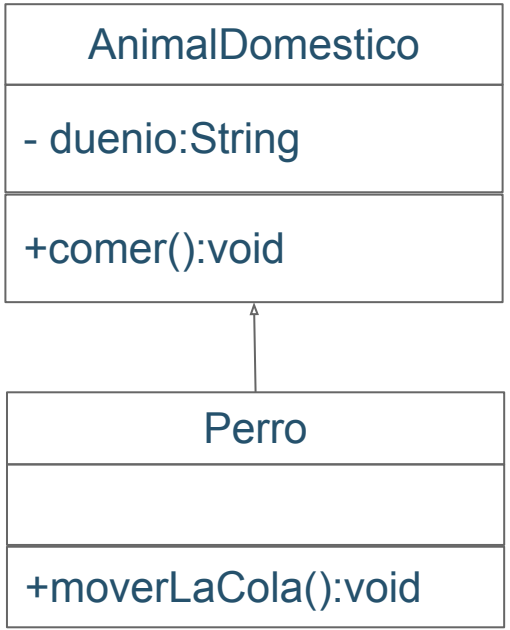
Veamos cómo diagramar herencias en UML.

un Perro

hereda de (es)

un AnimalDomestico

Esta relación se diagrama con una **línea sólida** y punta de flecha **vacía** desde el hijo hacia el padre.



UML - Herencia

En el diagrama anterior vemos que **Perro** *hereda* de **AnimalDomestico**. También se lee “Perro es un AnimalDomestico”.

En el diagrama se ven en **AnimalDomestico** el método **comer()**, y el atributo **duenio**. En **Perro**, no se vuelven a escribir, se entiende que los tiene por la relación de Herencia.

Se volverían a escribir, si específicamente fuera a modificarles el código.

Cuando un hijo sobreescribe el código de su padre, se llama **Override**.

En la clase **Perro**, sí escribimos el método **moverLaCola()**, porque es específico de **Perro**.