DigitalHouse>

Certified Tech Developer
The Ultimate Degree

Programación Orientada a Objetos

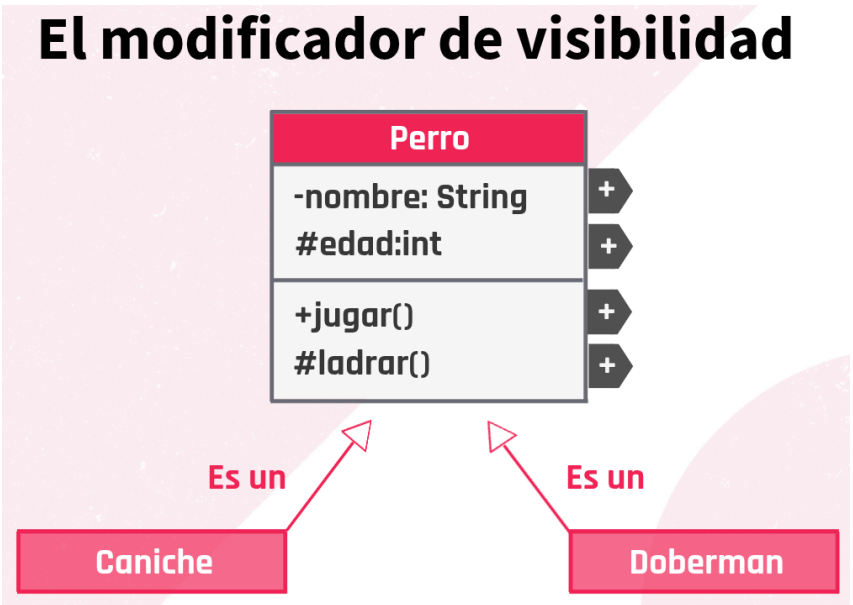
# Encapsulamiento y Herencia

*Recordemos que cuando una propiedad es pública significa que es accesible desde cualquier clase. Es decir, en el momento en que un objeto quiera acceder a un valor público puede obtenerlo y modificarlo sin ninguna operación de por medio. Esto sería el equivalente a no ocultar información y, por lo tanto, "romper" el encapsulamiento.*

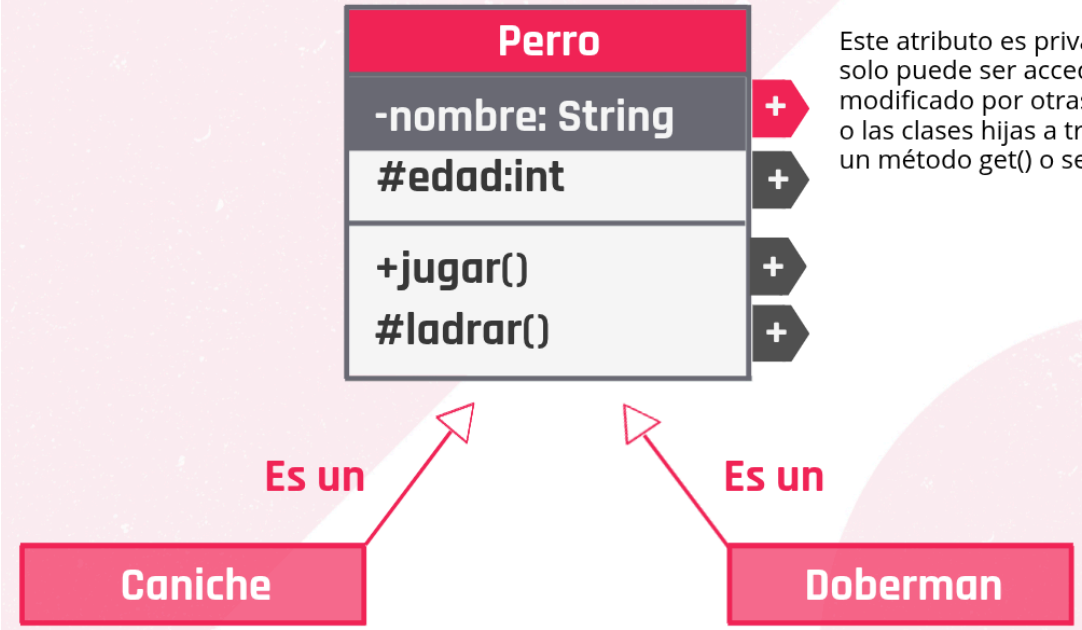
*Por el contrario, si declaramos un atributo privado limitamos completamente el acceso al dato. Nadie que no sea la propia clase puede acceder a ese dato. Siempre que se quiera acceder o modificar el dato, se debe hacer una operación para tal fin, por ejemplo, a través de getters o setters.*

*Con la herencia aparece un modificador de visibilidad nuevo llamado protegido, que en los diagramas UML se especifica con el "#", donde nos permite tener una visibilidad intermedia del atributo o método al que declaramos como tal. Es decir, es privado para otras clases, pero público para las clases hijas. El uso de este modificador de visibilidad "rompe" el encapsulamiento y evitaremos en lo posible su uso como buena práctica.*

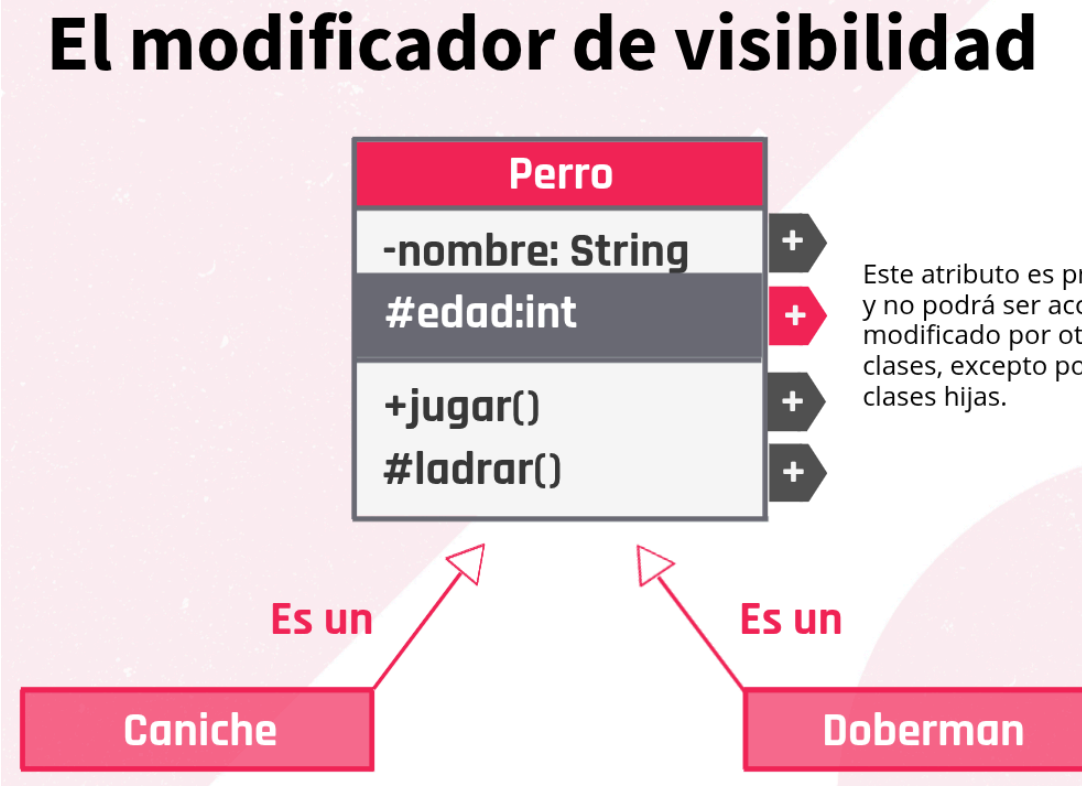
*Veamos esto en el siguiente ejemplo gráfico:*



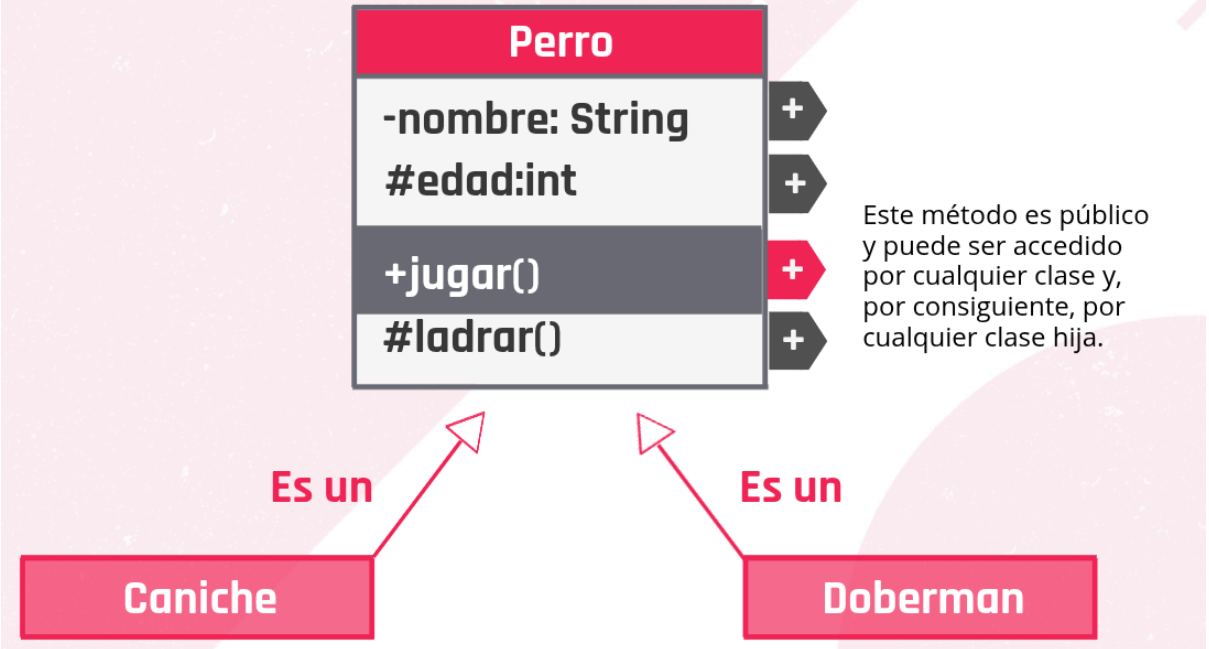
# El modificador de visibilidad



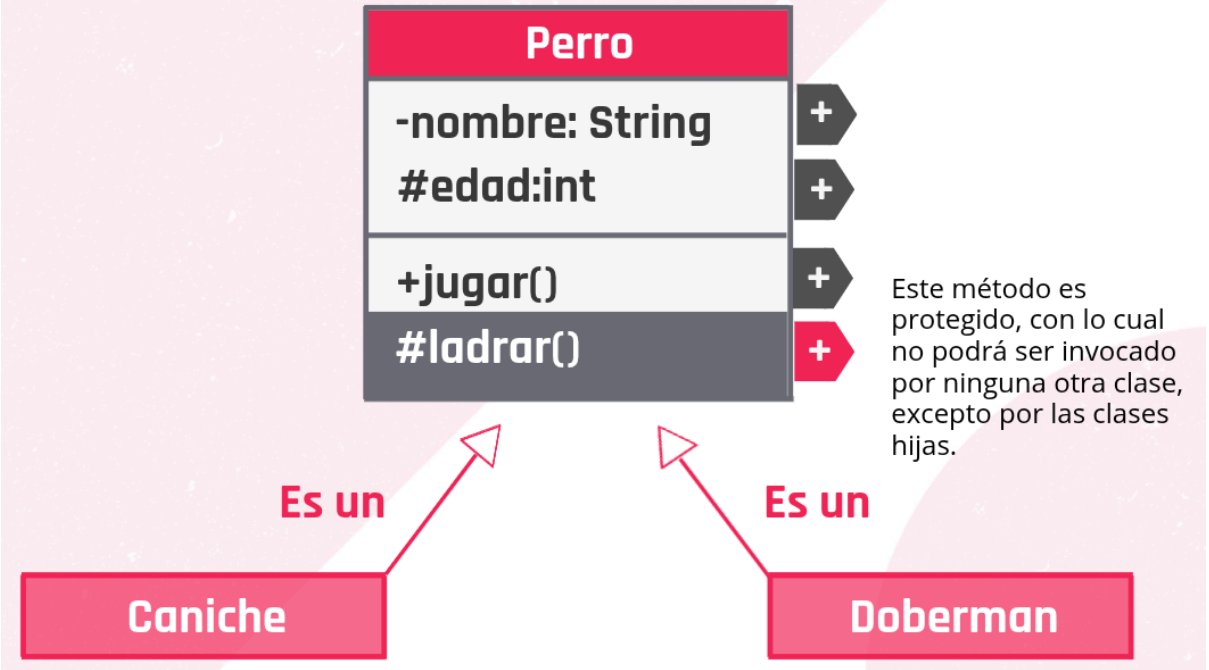

# El modificador de visibilidad



# El modificador de visibilidad




# El modificador de visibilidad



# *Firma de un método*

*En términos generales, una firma nos permite identificarnos y expresar nuestro consentimiento de un determinado documento. En nuestro documento de identidad podemos encontrar muchos elementos, entre ellos está la firma o rúbrica.*

*Con esta idea vamos a abordar la firma de un método en la programación orientada a objetos, que no es ni más ni menos que la definición completa de un método, es decir, su nombre, sus parámetros y sus tipos y el orden de aparición de dichos parámetros.*

*No podrán en una misma clase existir dos métodos con la misma firma, es decir, con el mismo nombre y cantidad de parámetros con sus respectivos tipos en el mismo orden. Decimos, entonces, que los siguientes métodos tienen diferentes firmas, son métodos diferentes porque, si bien se llaman igual, tienen diferente cantidad de parámetros o difiere alguno de sus tipos:*

*+ sumar(numero1: double, numero2: double): double*

*+ sumar(numero1: double, numero2: double, numero3: double): double*

*+ sumar(numero1: int, numero2: int): int*