Clases Abstractas en UML

DigitalHouse >
Coding School

Certified Tech Developer
The Ultimate Degree

Índice

1 | UML clase abstracta

Veremos **cómo se representa una clase abstracta** en nuestros diseños UML.

Clases abstractas

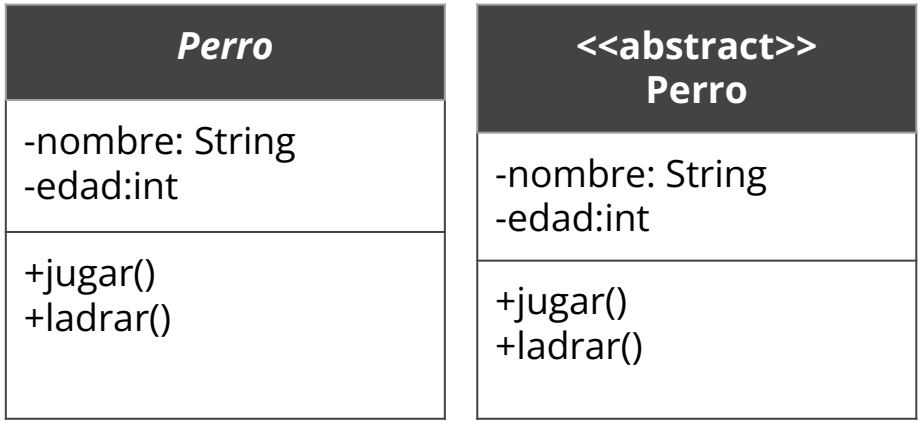
Las **clases abstractas** son aquellas que por sí mismas no se pueden identificar con algo “concreto” (no existen como tal en el mundo real), pero sí poseen determinadas características que son comunes en otras clases que heredarán de esta.

Estas clases abstractas nos permite declarar **métodos, pero que estos no estén implementados**, o sea, que no hacen nada en la clase abstracta, y estos métodos que también llamaremos abstractos obligarán a las subclases a sobreescribirlos para darles una implementación.

No olvidemos que no podremos instanciar objetos de una clase abstracta.

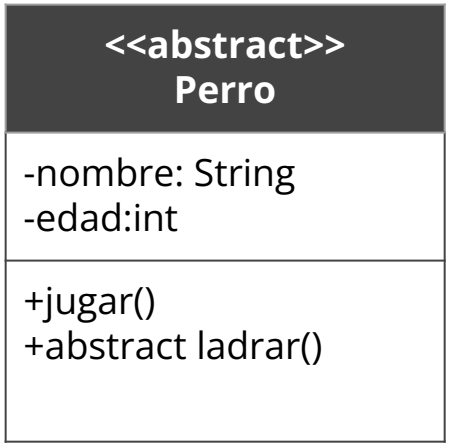
Clase abstracta en el diagrama UML

Las clases abstractas en el diagrama UML las representaremos ya sea indicando su nombre en **forma cursiva** o explicitando arriba de su nombre que es abstracta **<<abstract>>**, podemos optar cualquiera de las dos.

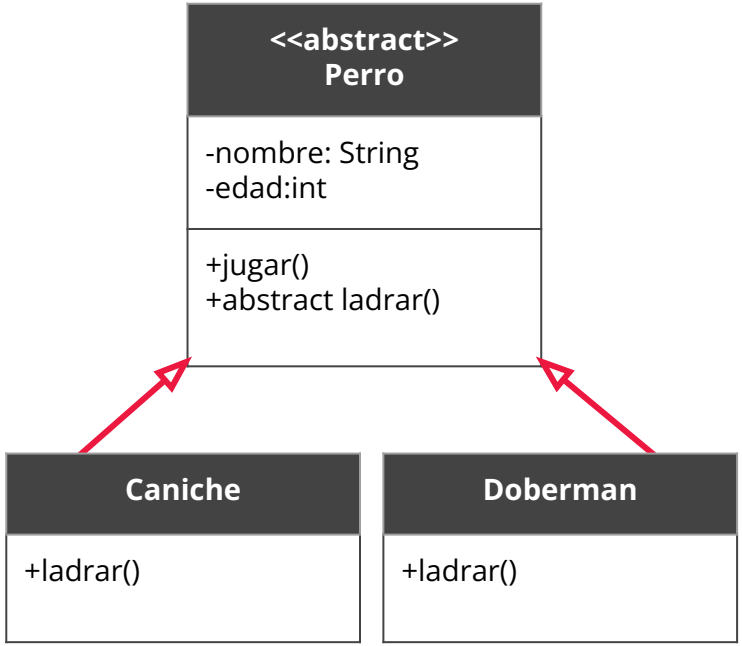


Los métodos abstractos los vamos a especificar en los diagramas UML como se muestra a continuación, el método ladrar será ahora un método abstracto y en los diagramas UML le anteponemos la palabra **abstract** para identificarlo como tal.

Al convertir en abstracto al método ladrar nos indica que el mismo **no está implementado en la clase Perro** y deberá implementarlo toda clase que herede de Perro.



En el ejemplo que vemos a continuación, al contar la clase abstracta Perro con un método abstracto, **obliga** a la clase Caniche y Doberman a **sobrescribir dicho método** implementandolo.



Todo Perro debe ladrar(), pero solo cada Perro específico sabe “cómo” debe ladrar.

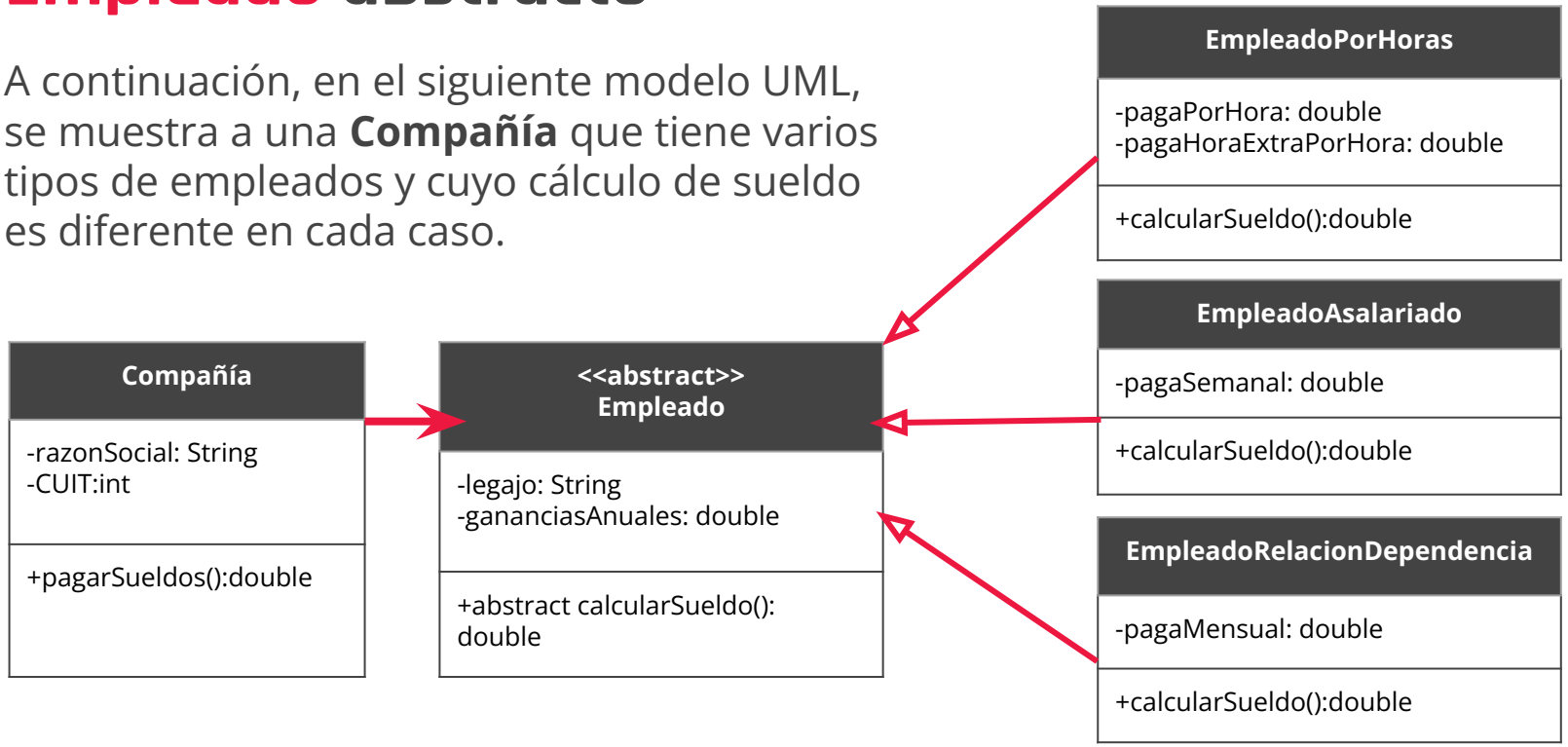
De esta manera una clase abstracta puede tener atributos y métodos que serán heredados por las subclases pero también puede contener **métodos abstractos que actúan como un contrato** obligando a estas subclases a implementar dichos métodos.

Si bien una clase abstracta puede tener uno o varios métodos abstractos, no es obligatorio que los tenga.

2 Ejemplos y buenas prácticas

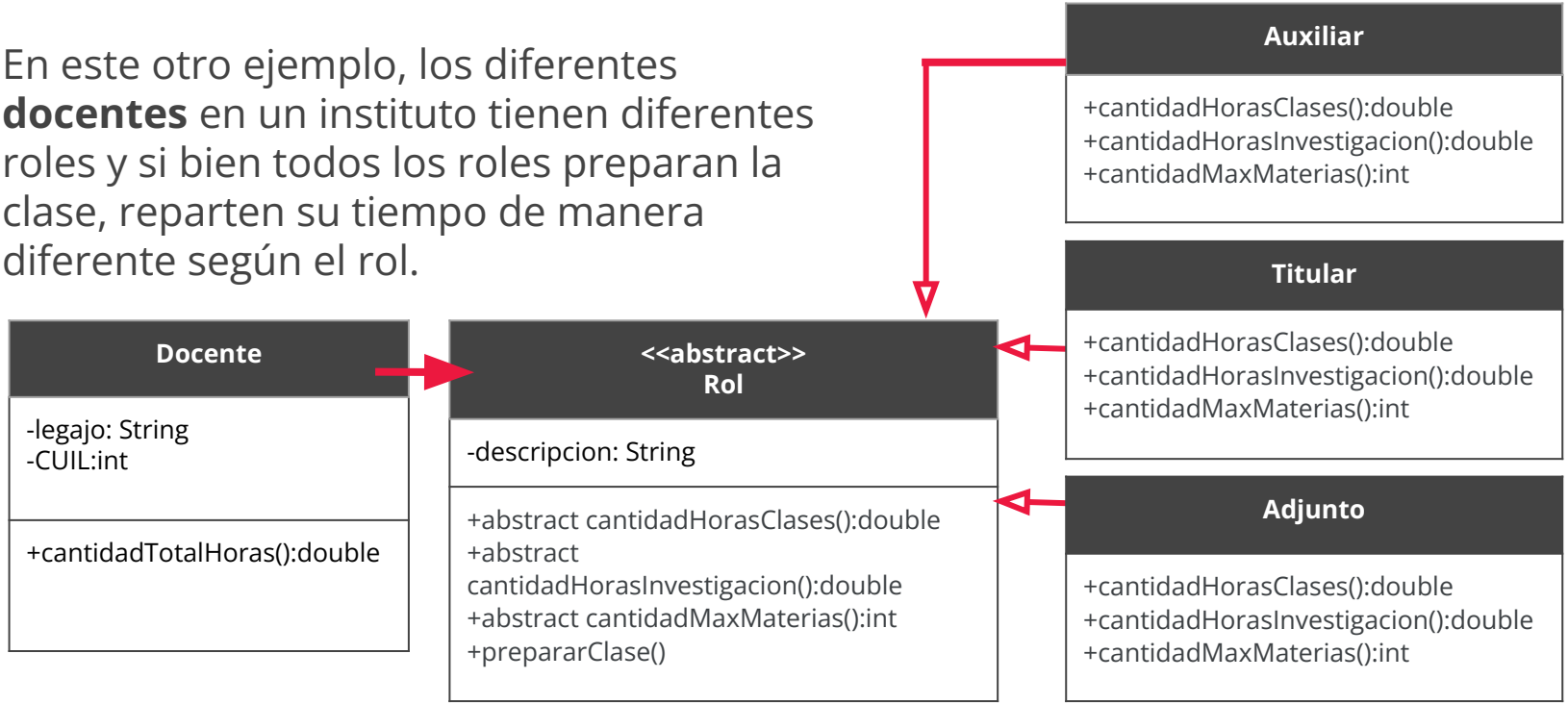
Empleado abstracto

A continuación, en el siguiente modelo UML, se muestra a una **Compañía** que tiene varios tipos de empleados y cuyo cálculo de sueldo es diferente en cada caso.



Rol abstracto

En este otro ejemplo, los diferentes **docentes** en un instituto tienen diferentes roles y si bien todos los roles preparan la clase, reparten su tiempo de manera diferente según el rol.



Martin Fowler describió a esta problemática de roles “Role Object” como un patrón de diseño para modelar roles, siendo una buena práctica a la hora de modelar esta problemática.


DigitalHouse>
Coding School