Diagramas de clases

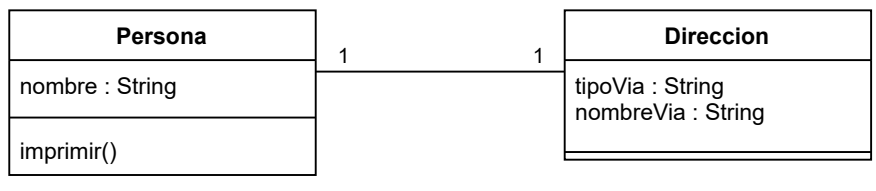
Un **diagrama de clases es una representación gráfica** de la vista estática del conjunto de clases que forman parte de un sistema, **junto con las relaciones existentes** entre ellas (Ver apuntes de clase del tema 5.2). Aquí hablaremos de las relaciones que podemos encontrar.

Podemos encontrar las diferentes relaciones entre las clases:

1) Asociaciones
2) Generalizaciones
3) Realizaciones
4) Dependencias

1) Asociaciones

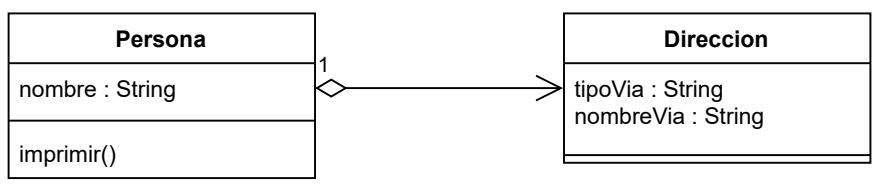
Es el tipo más general entre las relaciones entre clases y generalmente se representa en java poniendo un atributo de la otra clase con la que tiene relacion. Por ejemplo, si tenemos una asociacion de la clase Persona con la clase Direccion, entonces la clase persona tendrá un atributo de la clase Direccion. Además, este tipo de relaciones puede implicar bidireccionalidad, es decir que ambas clases tendrán un atributo de la otra. Siempre responde a la preguna **"Tiene un"** o en ingles **"has a"**. Se representa por **una linea oscura sin flechas**, pero pudiendo tener cardinalidad. Notese que en el diagrama de clases no hace falta poner el atributo de la otra clase



Dentro de las asociaciones podemos distinguir dos tipos, **composiciones o agregaciones**, que tiene que ver con el ciclo de vida de cuando se crea un objeto de un tipo

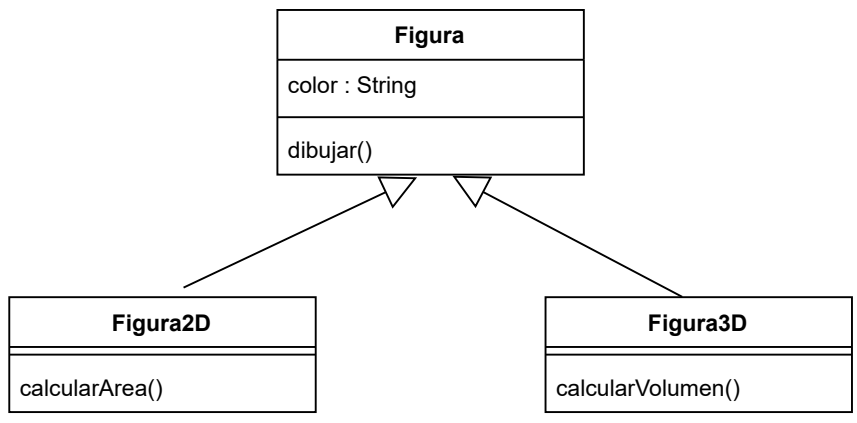
Composiciones, representadas por una flecha negra con un rombo negro -> Implica **dependencia fuerte** de un objeto con el otro, dicho de otra manera, cuando se crea un objeto, se crea tambien el otro (por ejemplo en el constructor). Lo podemos entender basicamente como que no tiene sentido la existencia de un objeto sin el otro.Tambien podemos poner cardinalidades. El objeto que lleva el ciclo de vida del otro es el que tiene el rombo negro (es el que tiene el atributo).

Agregaciones, representadas por una flecha negra con un rombo blanco -> Implica **dependencia debil** de un objeto con el otro, dicho de otra manera, cuando se crea un objeto NO tiene porque existir el otro, lo podemos crear más tarde para inyectarselo mediante un setter por ejemplo. Lo podemos entender basicamente como que el objeto A puede o no puede tener el objeto B en un instante de tiempo dado, pero tiene la capacidad para referenciarlo. Tambien podemos poner cardinalidades



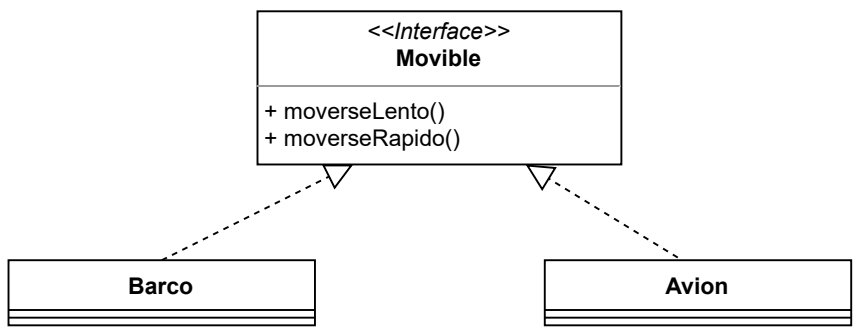
2) Generalizaciones

Este tipo de relaciones representa **herencia**, y se dan cuando queremos decir que una clase hereda o extiende de otra. Este tipo responde a la preguna **"Es un"** o ingles **"Is a"**. Se representa por una flecha continua con la punta blanca



3) Realizaciones o implementaciones

Este tipo de relaciones se dan cuando queremos decir que una clase implementa una interfaz. Se representa por una flecha discontinua con la punta blanca



4) Dependencia

Este tipo de relaciones representa cuando una clase usa a otra o tambien la instancia para hacer alguna tarea o para usar sus metodos, es decir, puede hace un **new** para crear un objeto o puede ser un parámetro de un método. Se representa por una flecha discontinua. Por ejemplo una clase main que instancie objetos de otras clases sería una relacion de dependencia