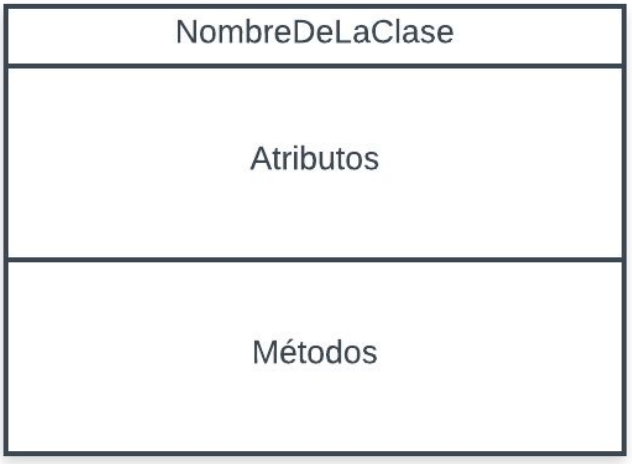
Este estándar está pensado para trabajar en los diagramas de clases dentro de los cursos de Digital House.

A. Especificación de una clase:

En nuestro diagrama, cada clase está representada por un rectángulo que contendrá tres divisiones:

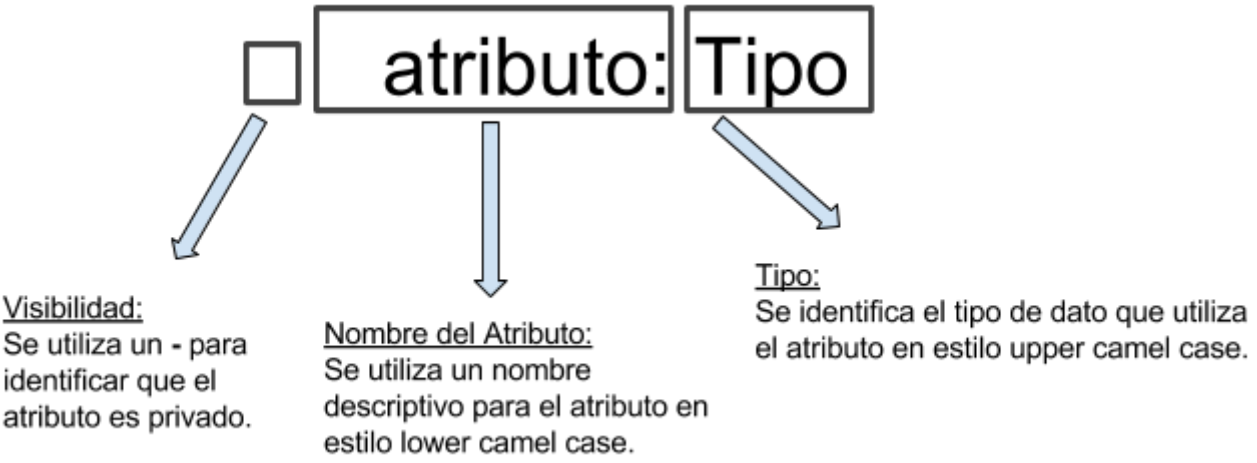
- ✓ Nombre de la clase.
- ✓ Atributos
- ✓ Métodos



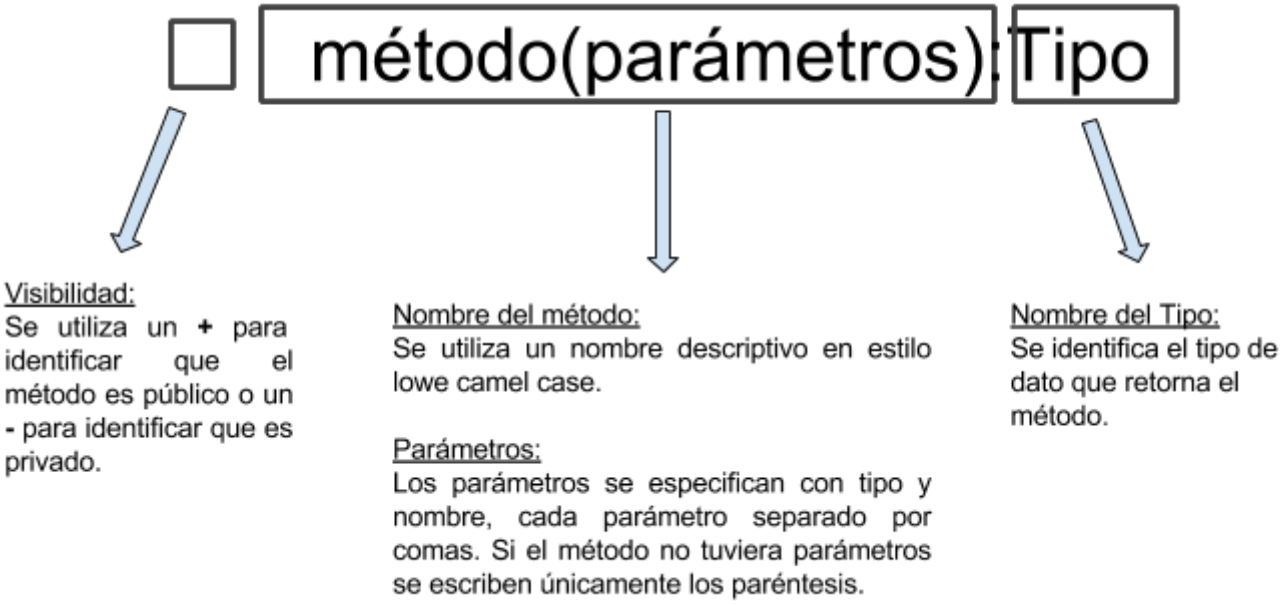
- **Especificación del nombre de una clase:** se deberá elegir un nombre descriptivo de la clase que se está diagramando. El nombre de la clase siempre comienza con mayúscula. Para nombres compuestos utilizamos el estilo de escritura upper camel case[1].

[1] **CamelCase** es un estilo de escritura que se aplica a frases o palabras compuestas.
UpperCamelCase, cuando la primera letra de cada una de las palabras es mayúscula.
Ejemplo: *EjemploDeUpperCamelCase*.
LowerCamelCase, igual que la anterior con la excepción de que la primera letra es minúscula.
Ejemplo: *ejemploDeLowerCamelCase*

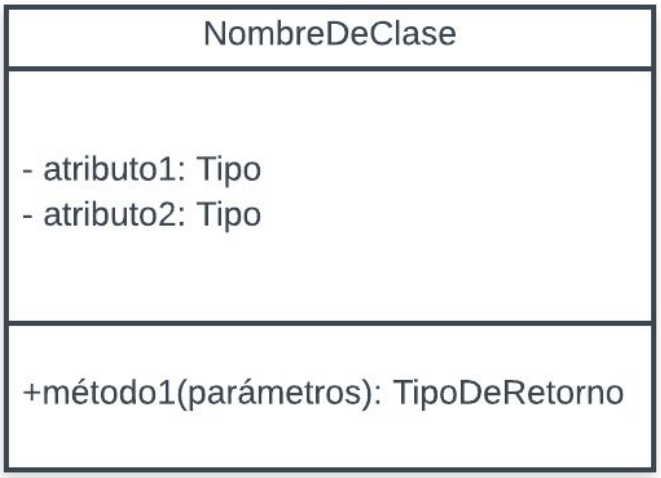
- **Especificación de los atributos:**



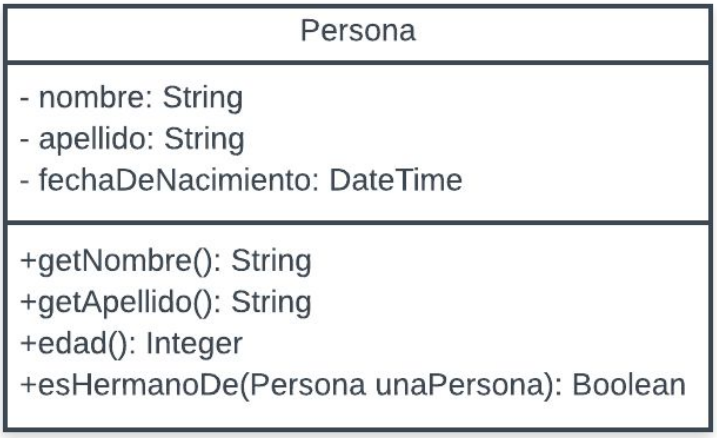
- **Especificación de los métodos:**

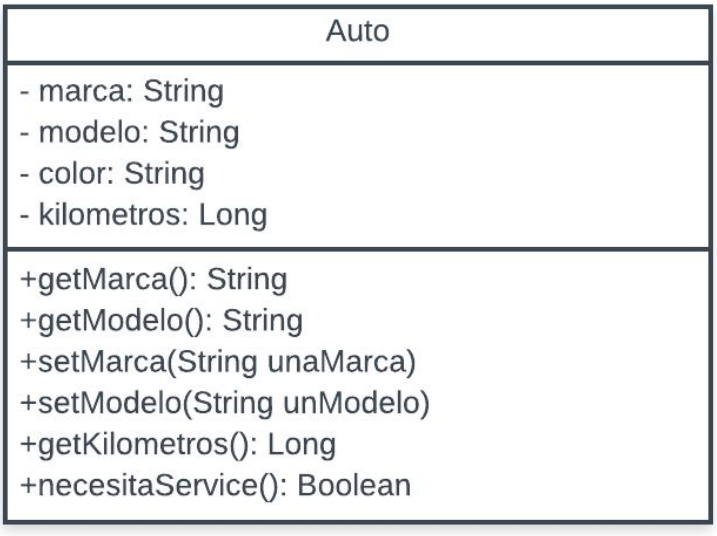


Esquema



Ejemplos





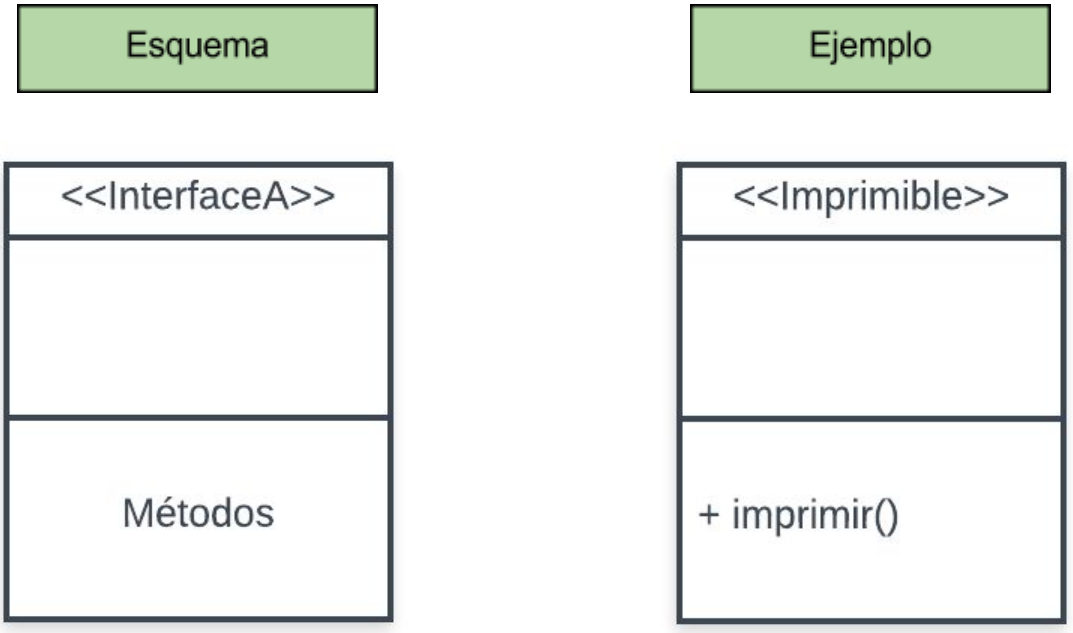
B. Especificación de una interfaz:

En nuestro diagrama, una interfaz se especifica de forma similar a una clase, pero sin atributos:

- ✓ Nombre de la interfaz.
- ✓ Métodos

- **Especificación del nombre de una interfaz:** se deberá elegir un nombre descriptivo de la interfaz que se está diagramando. El estilo utilizado es upper camel case al igual que en el nombre de una clase. Además utilizamos los símbolos << >> para envolver al nombre (Ver esquema y ejemplo).

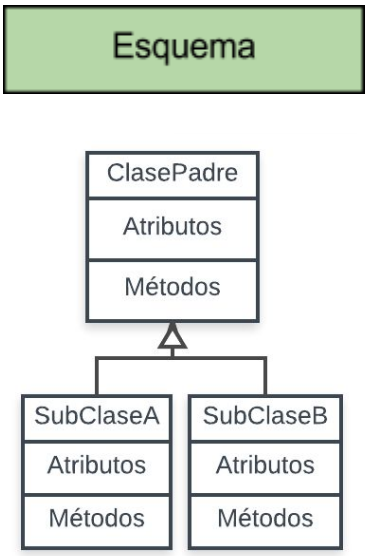
- **Especificación de los métodos de una interfaz:** los métodos en una interfaz, por definición, son públicos por lo tanto su visibilidad siempre será **+**

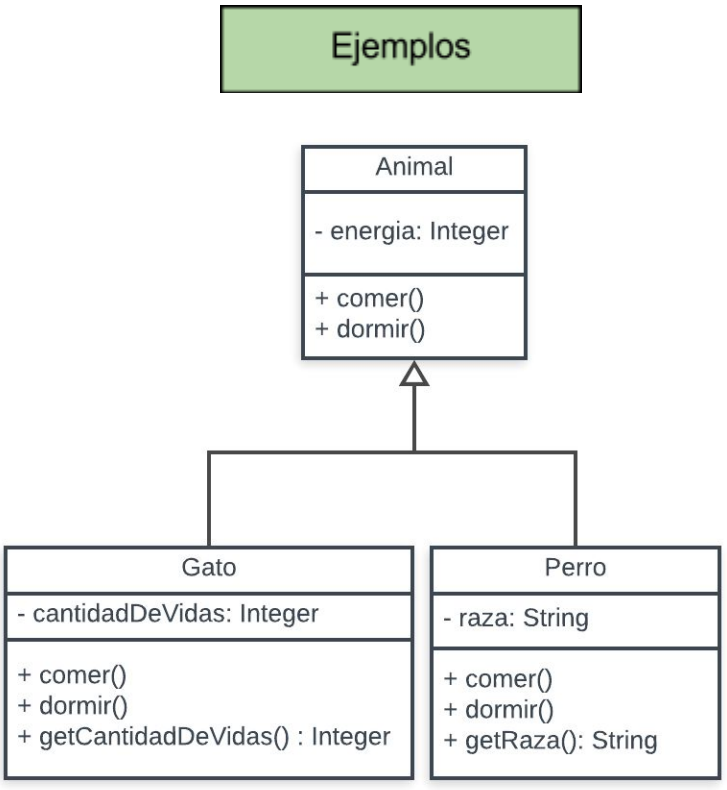


C. Especificación de las relaciones entre clases:

A continuación vamos a describir cómo se especifican, según nuestro estándar, las diversas relaciones que pueden existir entre las clases.

- **Especificación de la relación de herencia:** una clase puede tener un único padre. La subclase hereda los atributos y los métodos de su clase padre.

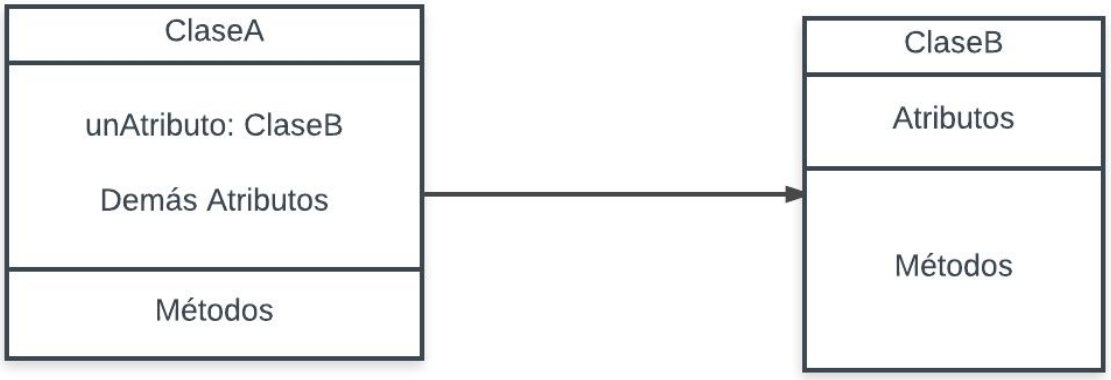




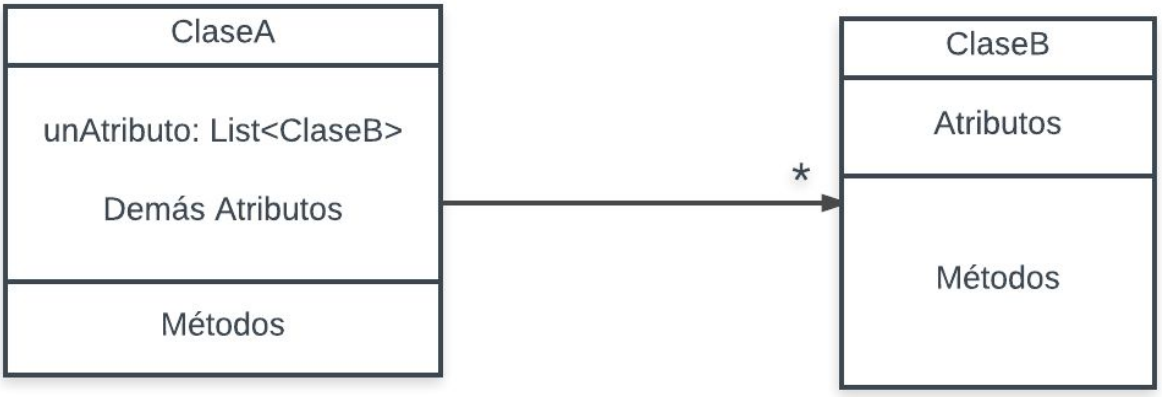
- **Especificación de la relación de asociación:** La relación de asociación se establece cuando un objeto de una clase colabora con uno o más objetos de otra clase. En el standard UML existen diversos tipos de relaciones de asociación y composición, sin embargo nosotros especificaremos una sola. Por otro lado, a la relación le especificaremos su aridad, es decir con cuantos objetos de la otra clase se relaciona.

Esquema

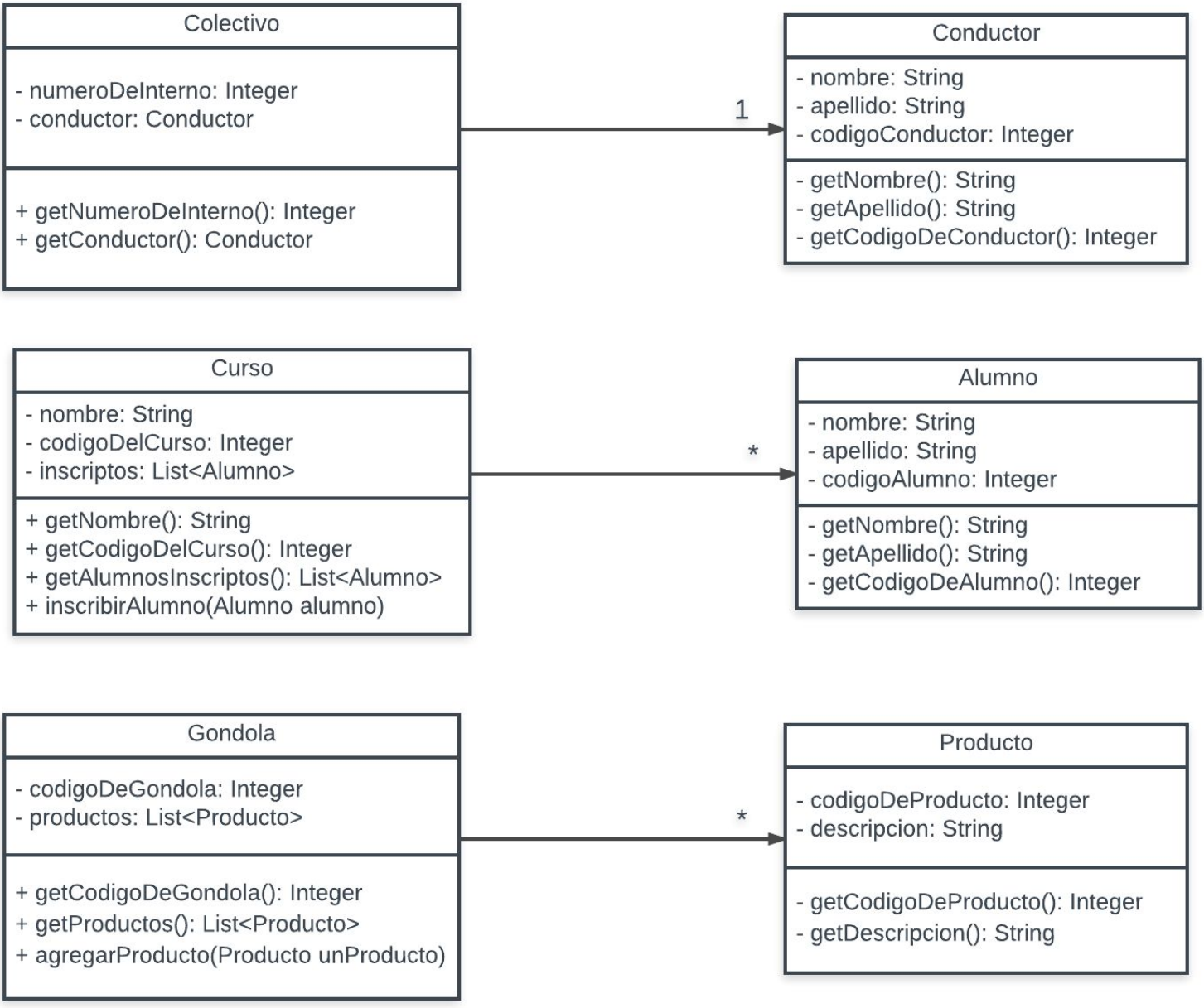
En este diagrama diremos que la **ClaseA** conoce a **una instancia** de la **ClaseB**. Sin embargo, debemos tener en cuenta que, en este caso, la **ClaseB** no conoce a la **ClaseA**.



En este diagrama diremos que la **ClaseA** conoce a **varias instancias** de la **ClaseB**. Sin embargo, debemos tener en cuenta que, en este caso, la **ClaseB** no conoce a la **ClaseA**.

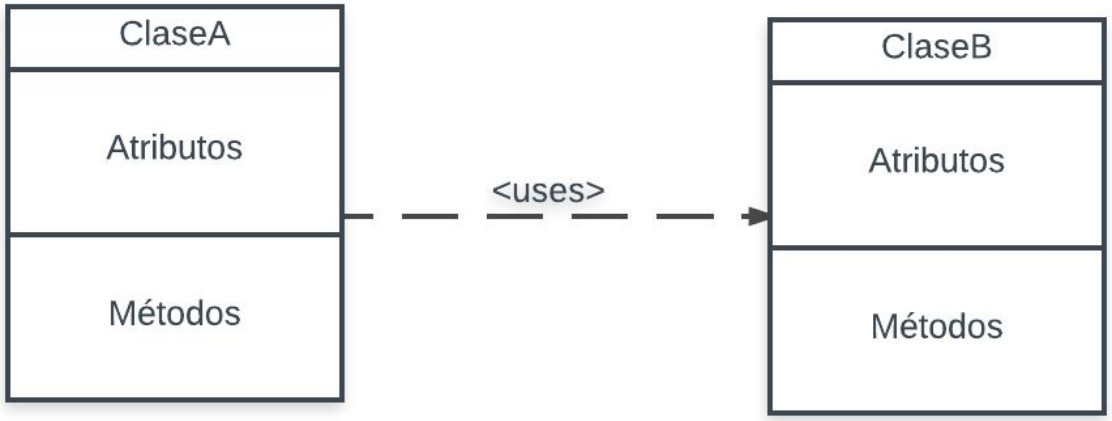


Ejemplos

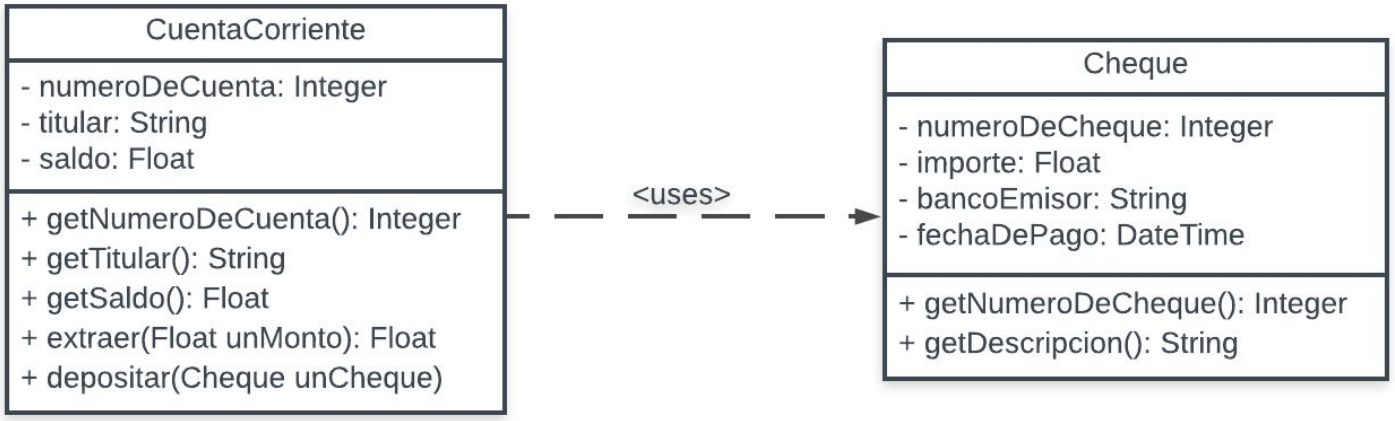

Colectivo
- numeroDeInterno: Integer
- conductor: Conductor
+ getNumeroDeInterno(): Integer
+ getConductor(): Conductor
1
Conductor
- nombre: String
- apellido: String
- codigoConductor: Integer
- getNombre(): String
- getApellido(): String
- getCodigoDeConductor(): Integer
Curso
- nombre: String
- codigoDelCurso: Integer
- inscriptos: List<Alumno>
+ getNombre(): String
+ getCodigoDelCurso(): Integer
+ getAlumnosInscriptos(): List<Alumno>
+ inscribirAlumno(Alumno alumno)
*
Alumno
- nombre: String
- apellido: String
- codigoAlumno: Integer
- getNombre(): String
- getApellido(): String
- getCodigoDeAlumno(): Integer
Gondola
- codigoDeGondola: Integer
- productos: List<Producto>
+ getCodigoDeGondola(): Integer
+ getProductos(): List<Producto>
+ agregarProducto(Producto unProducto)
*
Producto
- codigoDeProducto: Integer
- descripcion: String
- getCodigoDeProducto(): Integer
- getDescripcion(): String

- **Especificación de la relación de uso:** La relación de uso se establece cuando un objeto de una clase utiliza algún objeto de otra clase de nuestro modelo. En general, las relaciones de uso se establecen cuando en algún método se utiliza un objeto de otra clase.

Esquema



Ejemplos



- **Especificación de la relación de implementa:** La relación de implementa se establece cuando una clase implementa una interfaz.