

P00 y Conceptos Básicos de las Estructuras de Datos

Luis Gerardo Montané Jiménez

Agosto 2017



Introducción

- ▶ El término Orientado a Objetos (OO) promueve que el software sea organizado como una colección de objetos que contienen datos y comportamientos
- ▶ Se busca hacer que el software sea más fácil de mantener, escribir y reutilizar
- ▶ Las características básicas de la programación OO (POO) son: Abstracción, Encapsulación, Polimorfismo y Herencia

Abstracción (1/2)

- ▶ La abstracción en la POO promueve el modelado centrado en aspectos esenciales de una entidad, ignorando sus propiedades no relevantes
- ▶ En la construcción de software significa centrarse en lo que es y lo que hace un objeto antes de decidir cómo debería ser implementada
- ▶ Para apoyar la construcción de sistemas bajo el paradigma OO han surgido modelos que ayudan la abstracción de un problema

Abstracción (2/2)

- ▶ El uso de modelos para la programación OO tiene como finalidad la abstracción de aquellos aspectos que sean importantes
- ▶ Un buen modelo Orientado a Objetos (OO) captura los aspectos cruciales del problema y omite los demás
- ▶ Un modelado de objetos captura la estructura estática del sistema
- ▶ El modelo de clases corresponde con el mundo real de manera más fiel

Clase

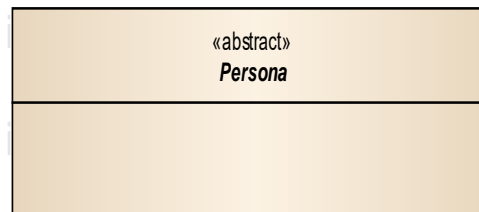
- ▶ Es una abstracción que permite definir un tipo de objeto, junto con propiedades (atributos) y operaciones (métodos)
- ▶ Es un elemento para la creación de objetos a partir de un modelo pre-definido

Objeto

- ▶ Entidad existente que tiene propiedades con datos del mismo objeto y operaciones específicas (métodos)
- ▶ Es el resultado de instanciación de una clase

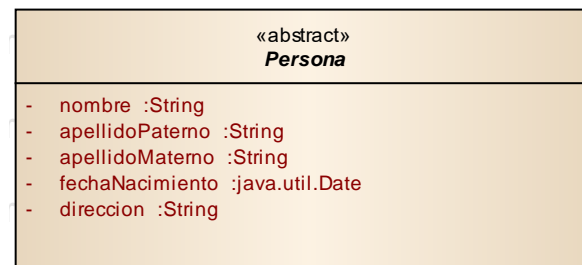
Encapsulación

- ▶ La encapsulación es un mecanismo de programación por el que se establece una relación entre las operaciones y los datos que se manipulan
- ▶ En los lenguajes de programación OO es posible relacionar los datos y las operaciones en cajas negras independientes
- ▶ Dentro de un objeto los datos o las operaciones pueden ser privados o públicos
- ▶ El código privado únicamente es accedido desde adentro del objeto, mientras que lo público se puede acceder desde otro objeto
- ▶ La unidad básica de encapsulación es la clase



Atributos y Propiedades

- ▶ Un atributo es un valor de un dato que está almacenado en los objetos de una clase, ejemplo de atributos son:
 - ▶ Nombre y fecha de nacimiento
- ▶ Cada atributo tiene un valor para cada instancia del objeto
- ▶ Las instancias distintas de una cierta clase pueden tener el mismo valor o valores distintos para un atributo dado



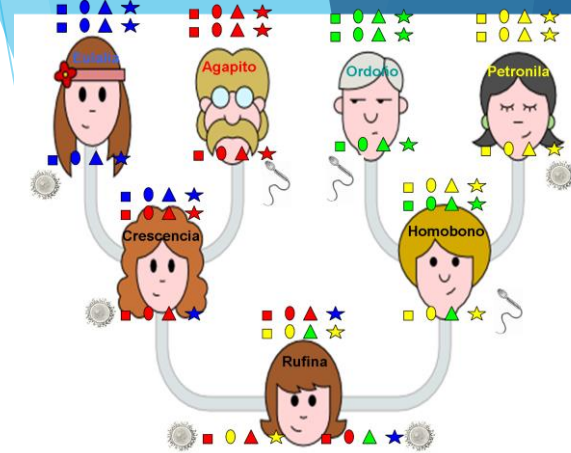
Operaciones

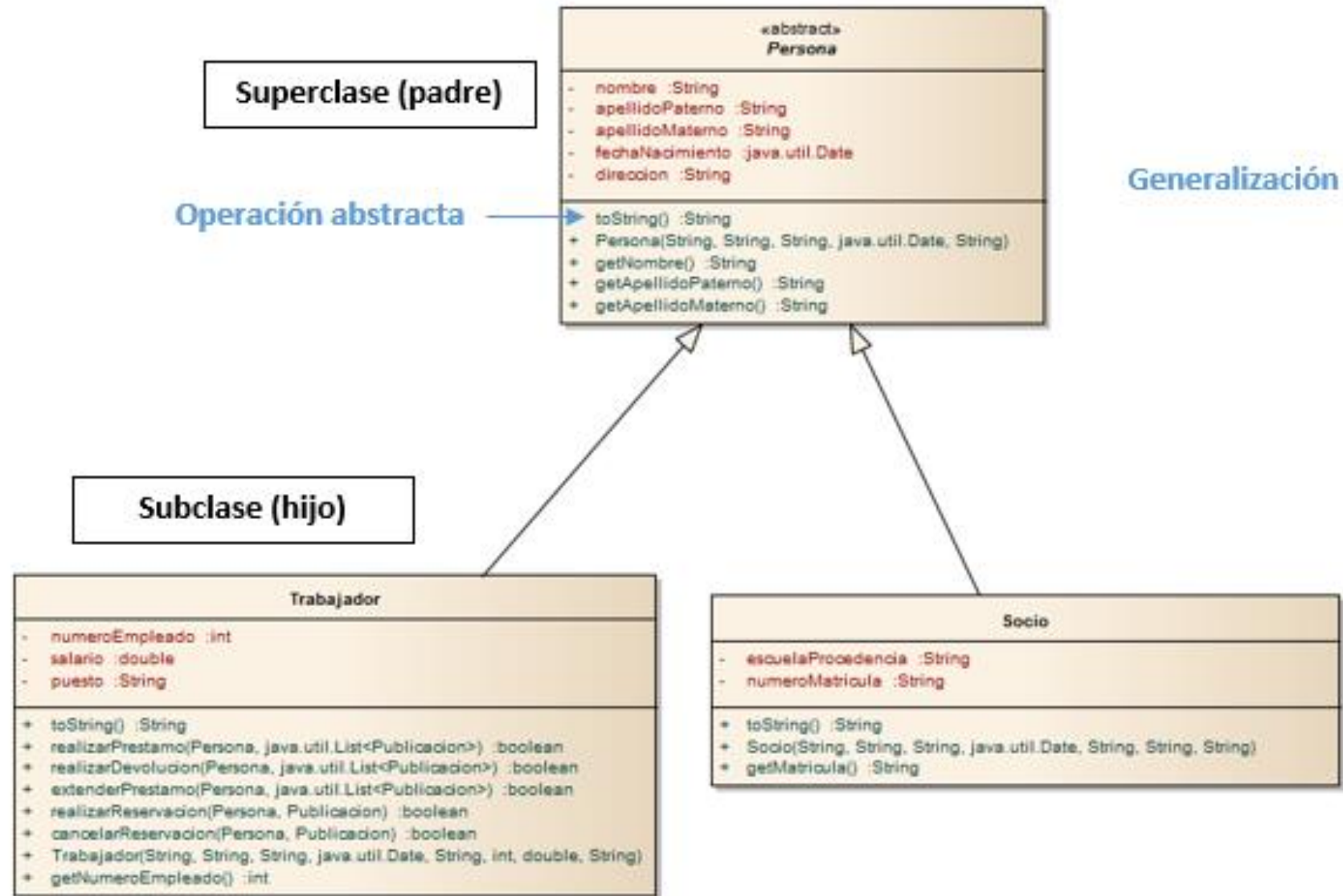
- ▶ Una operación es una función o transformación que se puede aplicar o que puede ser aplicada por los objetos de una clase
- ▶ Por ejemplo, prestar, devolver y reservar son operaciones de la clase Trabajador o Biblioteca
- ▶ Todos los objetos de una clase comparten las mismas operaciones

«abstract» <i>Persona</i>	
-	nombre :String apellidoPaterno :String apellidoMaterno :String fechaNacimiento :java.util.Date direccion :String
+	toString() :String Persona(String, String, String, java.util.Date, String) getNombre() :String getApellidoPaterno() :String getApellidoMaterno() :String

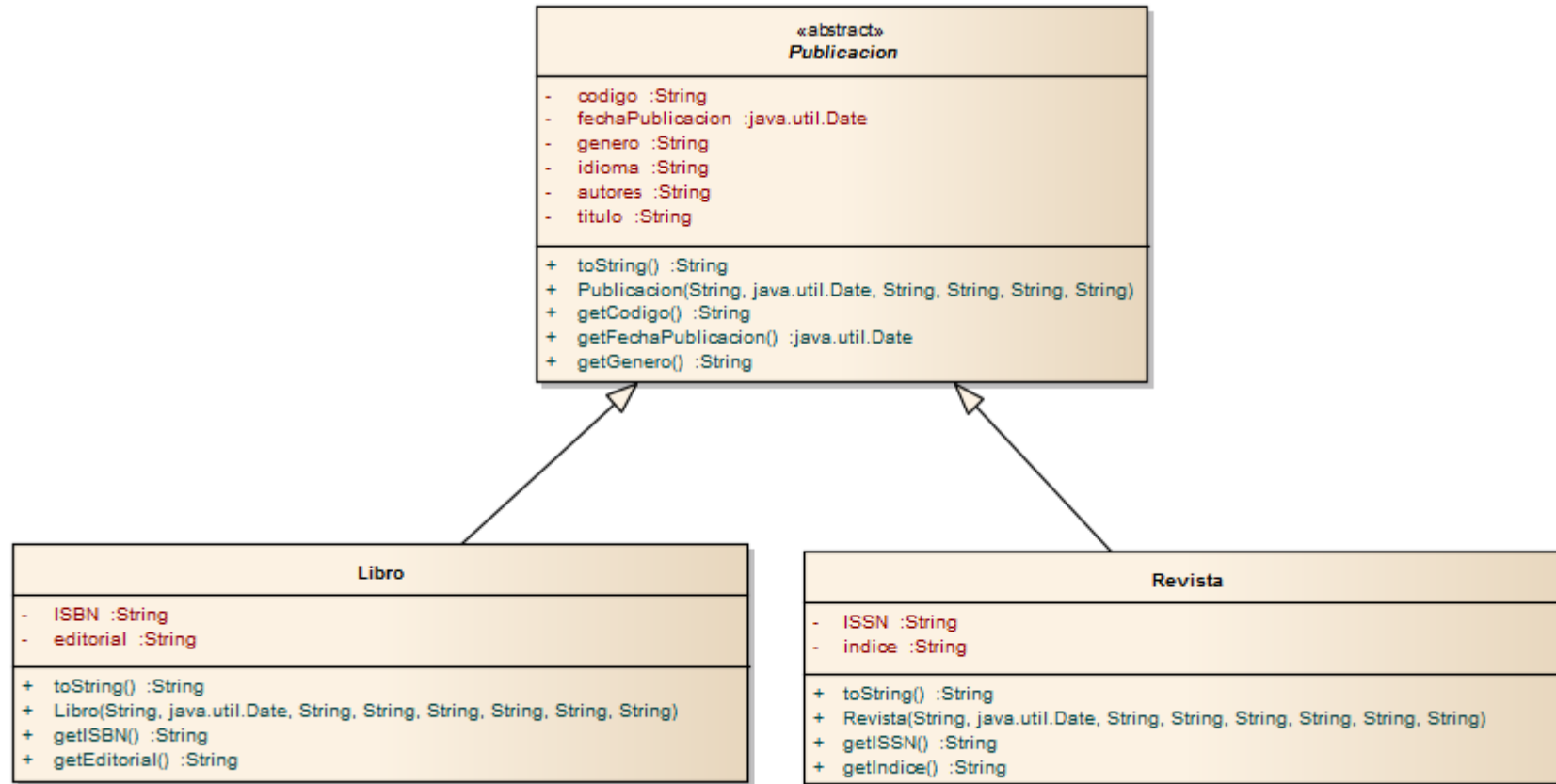
Herencia (1/3)

- ▶ La herencia es el proceso mediante el cual un objeto puede adquirir propiedades de otro
- ▶ La clase hereda las propiedades generales de su padre
- ▶ La herencia es el mecanismo que le permite a un objeto ser una instancia específica de una clase más general
- ▶ En este ámbito, se introducen los términos de *subclases* y *superclases*
- ▶ Las subclases contiene los atributos y métodos de la clase de la cual se deriva (superclase)
- ▶ La herencia es una potente abstracción para compartir similitudes entre clases
- ▶ Puede representarse visualmente de forma jerárquica, comenzando con la clase base llamada también superclase de la cual se derivan las clases secundarias





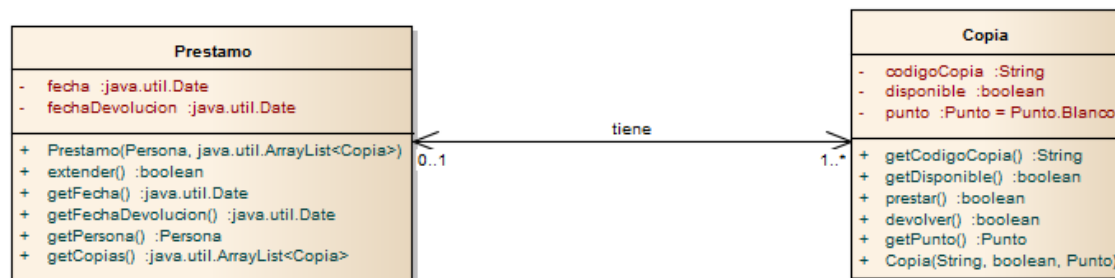
Herencia (3/3)



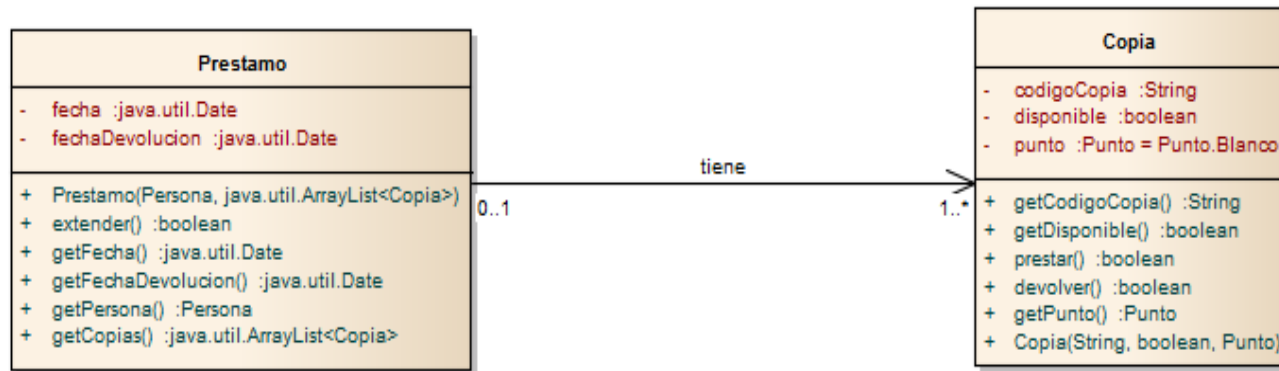
Asociaciones Bidireccionales y Unidireccionales

- ▶ Los enlaces muestran una relación entre dos (o más objetos)
- ▶ El modelado de un enlace oculta el hecho consistente en que el enlace no forma parte ninguno de los objetos por sí mismo, sino que depende de ambos a la vez

Asociaciones Bidireccionales

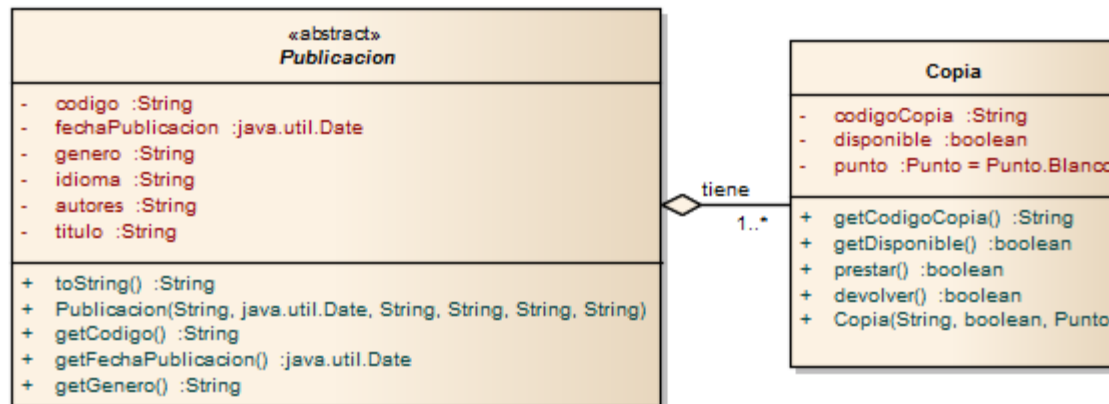


Asociación Unidireccional



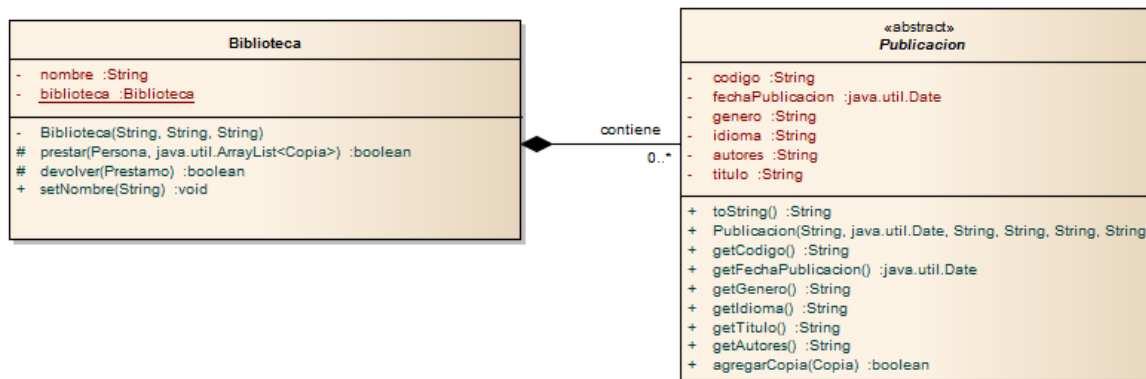
Agregación

- ▶ La agregación es la representación de una relación
- ▶ Para su representación se utiliza un diamante hueco en el extremo de la trayectoria unida a la clase agregada
- ▶ La vida del objeto agregado no depende del objeto compuesto



Composición

- ▶ Una composición es una forma más fuerte de asociación
- ▶ El objeto compuesto es el responsable único de gestionar sus partes
- ▶ En java, esta composición se puede implementar con una lista de *Publicaciones* agregada en la clase *Biblioteca*



Contenido

- ▶ Abstracción
- ▶ Encapsulación
- ▶ Herencia
- ▶ **Diagrama de Clases**

Diagrama de Clases

