

II. Diseño y Modelado de Clases

1. *Clases, Objetos, Sobrecarga, Sobre-escritura*

Luis Gerardo Montané Jiménez

Septiembre 2017

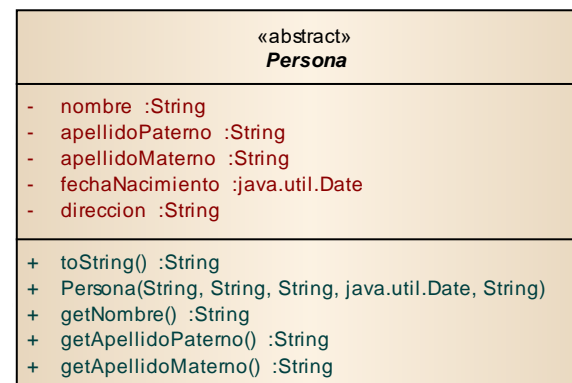


Objetivo

- ▶ Abordar conceptos de sobre-carga de métodos, constructores y sobre-escritura de métodos

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

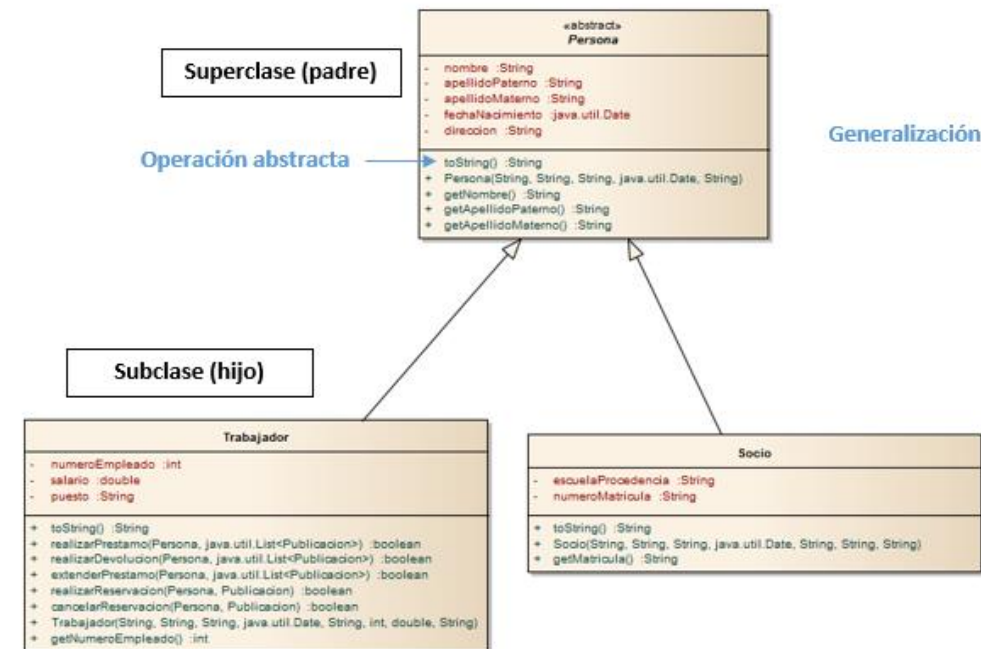


Objeto

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

Alcance de la Herencia

- ▶ La herencia es transitiva: una clase puede heredar características de superclases que se encuentran muchos niveles más arriba en la jerarquía de herencia
- ▶ La herencia es clasificada con la dimensión *variedad*:
 - ▶ A ES-UN B, y donde la clase A se relaciona con B por herencia
 - ▶ Ejemplos:
 - ▶ Un Trabajador es una Persona
 - ▶ Un Socio es una Persona
 - ▶ Un Perro es un Animal
 - ▶ Un Auto es un Vehículo
 - ▶ Un Motor es un Vehículo ¿Incorrecto?



Implementación en Java

- ▶ Sobre-escritura de métodos
 - ▶ Consiste en cambiar el comportamiento de un método que tiene la misma firma en una clase derivada
- ▶ Si el método de la clase base se pretende que no sea sobre-escrito entonces el método debe ser declarado utilizando la palabra reservada **final**
- ▶ Ejemplo:
 - ▶ Si la clase Persona es la clase base (superclase, padre) y se busca tener una clase derivada entonces la herencia se realiza de la siguiente forma:
 - ▶ Superclase: `class Persona{....}`
 - ▶ Clase derivada (subclase): `class Trabajador extends Persona{....}`
 - ▶ Clase derivada (subclase): `class Socio extends Persona{....}`
 - ▶ Para acceder a elementos de la clase base se utiliza la palabra reservada **super**.

Sobre-carga de métodos (1 / 3)

- ▶ La firma de un método es la combinación del tipo de dato que regresa, su nombre y lista de parámetros
- ▶ La sobre-carga de métodos es la creación de varios métodos con el mismo nombre pero con diferentes firmas (nombre del método, cantidad, tipo y orden de parámetros)

```
int calcular(int x, int y, int z){  
    ...  
}  
  
int calcular(double x, double y, double z){  
    ...  
}
```

- ▶ El tipo de valor de retorno no forma parte de la firma del método (no se utiliza para distinguir entre métodos)

```
int calcular(int x, int y, int z){...}  
double calcular(int x, int y, int z){...}
```

Sobre-carga de métodos (2/3)

- ▶ Los métodos sobrecargados poseen el mismo nombre sin importar el numero de métodos que existan
- ▶ Se puede usar cualquier tipo de método (String, int, float, double, etc....)
- ▶ En caso de que el método sea diferente de **void** se debe de retornar un valor dependiendo del tipo de método declarado
- ▶ Los parámetros o argumentos que posean los métodos sobrecargados pueden ser de diferentes tipos y diferente cantidad de estos

Sobre-escritura de métodos (3/3)

- ▶ La subclase o clase derivada hereda todos los métodos de su superclase que son accesibles a dicha subclase a menos que sobrescriba los métodos
- ▶ La subclase sobrescribe un método de la superclase cuando define un método con la misma firma o características (nombre, número y tipo de parámetros) que el método de la superclase
- ▶ La sobrescritura de métodos en las subclases es utilizada tradicionalmente para agregar o modificar la funcionalidad del método heredado de la clase padre

Sobre-carga de constructores

- ▶ La sobrecarga de constructores es cuando en una clase existen constructores múltiples

```
Constructor()  
{  
    nombre = null;  
    edad = 0;  
    direccion = null;  
}
```

```
Constructor(String nombre, int edad, String direccion)  
{  
    this.nombre = nombre;  
    this.edad = edad;  
    this.direccion = direccion;  
}
```

Diagrama de Clases

