ELO329 - Diseño y Programación Orientados a Objetos

Interfaces

(no interfaz gráfica)

Agustín González
Patricio Olivares

¿Qué queremos decir con interfaces?

- Aquí el término interfaz **NO** se refiere a las interfaces gráficas (esas ya vienen...)
- Una **interfaz**, al igual que una clase, declara el comportamiento esperado para objetos; sin embargo, a diferencia de una clase, no se especifica el cómo. Es decir, no incluye la implementación de cada método.
- Entre otros, interfaces y clases internas son recursos esenciales en el manejo de interfaces gráficas en Java. Éste será el próximo tópico.

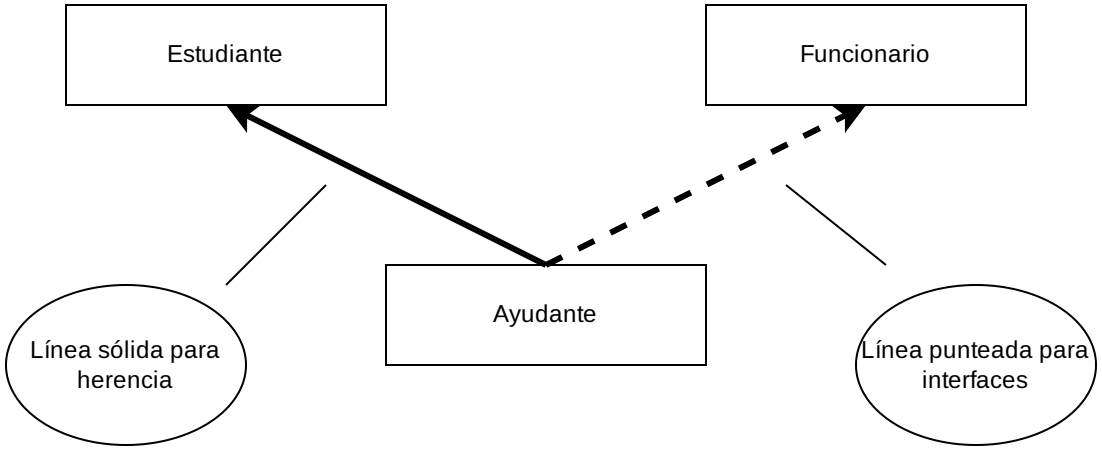
Nota linguística:

- Faz <=> cara; "entre caras" => interfaz, plural interfaces (femenino)
- Fase <=> etapa; entre dos fases => interfase, plural interfases.

Interfaces

- Una **interfaz** es la descripción de uno o más servicios o comportamientos (métodos) que posteriormente alguna clase puede implementar (y por ende ofrecer).
- Por ejemplo, si un estudiante es ayudante, entonces podríamos preguntarle por su sueldo. Un ayudante además de ser estudiante es capaz de responder consultas propias de un funcionario. Así un Ayudante, además de ser Estudiante (herencia), cumple con la interfaz Funcionario. También podríamos decir que él **es un** "Funcionario" (la misma relación que en herencia).
- **Gracias a las interfaces podemos crear métodos genéricos (como ordenar) que manipulan objetos y les exijan a éstos solo funcionalidades mínimas requeridas (como poder compararse).**

Representación gráfica UML



- Algunos lenguajes como C++ permiten herencia múltiple. Java permite heredar de **solo una clase base**.
- Para codificar situaciones en que herencia múltiple podría ser necesaria, Java hereda de una clase e implementa interfaces para las otras.
- **La relación de subtipo y "es-un" también deben cumplirse con interfaces**.

Interfaces: aspectos operativos

1° Paso - **Definir la interfaz:** Incluir solo los prototipos de los métodos de la interfaz.

→ Como resultado tenemos un archivo que define la interfaz.
Ej. Funcionario.java

2° Paso - **Implementación de la interfaz:** cada método **debe** ser implementado en alguna clase. Para cada método ponemos su prototipo y una implementación.

→ Como resultado tenemos una clase que implementa todos los métodos de la interfaz Funcionario, además de los propios de la clase.
Ej. Ayudante.java

Interfaces: Ejemplo 1

1. Definición de la interfaz en archivo `Funcionario.java`:

```
public interface Funcionario {
    int getSalary();
}
```

2. Implementación de la interfaz en clase `Ayudante.java`.

```
class Ayudante extends Estudiante implements Funcionario {
    public Ayudante(String name, String major, int s) {
        super(name, major); // Un constructor en Estudiante
        salary =s;
    }
    public int getSalary() { // implementamos la interfaz
        return salary; // podrían haber varios métodos
    }
    private int salary;
}
```

Interfaces: Ejemplo 2

1. Debemos atender dos cosas:
    - Si la interfaz no existe en el lenguaje, la debemos definir.
    - Definición de una interfaz, en un archivo de nombre `Comparable.java`, poner:

```
public interface Comparable {
    int compareTo (Object other);
} // Esta interfaz está ya definida en java.lang
```

Interfaces: Ejemplo 2

2. Luego debemos implementar la interfaz en alguna clase.
    - Implementación de una interfaz:

```
class Employee implements Comparable {
    ....
    public int compareTo(Object other) {
        ....// implementación
    }
}
```

Ejemplo 3: uso de interfaces

- Consideremos la extensión de la clase `Employee` para ordenar arreglos de empleados según su salario.
- La interfaz `Comparable` ya está definida en el lenguaje, luego solo debemos implementarla.
- Ver `EmployeeSortTest.java`.
- Ver documentación de clase Arrays e interfaz Comparable. Notar el método estático `sort(Object[ ] a)` de la clase `Arrays`.

Ejemplo 3: uso de interfaces

- No se permite crear instancias (objetos) de una Interfaz por la misma razón que no se puede crear instancias de clases abstractas: *No se tienen implementaciones*.

```
new Comparable(); // es un error!
```

Interfaces (cont.)

- En Java cada clase puede heredar de solo una clase, pero puede **implementar varias interfaces**.
- Cuando existe relación es-un con varias categorías del mundo real, usamos herencia con una de ellas e interfaces para exhibir el comportamiento esperado para las otras.
- **Todos los métodos de una Interfaz son públicos**. No es necesario indicarlo.
- Pueden incluir **constantes**. En este caso **son siempre** `public` `static` `final`.
- Se cumple también el principio de sustitución. Instancias de la clase que implementa una Interfaz pueden ser referenciadas por una referencia a la interfaz.

Otro ejemplo de Interfaz: Timer

- Java ofrece tres clases `Timer`. Una de ellas es `javax.swing.Timer`.
- Este `Timer` permite invocar un método periódicamente y en paralelo a la ejecución de main
- Constructor: `Timer(int delay, ActionListener listener)`
  - `delay` tiempo entre invocaciones del método `listener.actionPerformed()`.
  - La clase de listener debe implementar la interfaz `ActionListener`, la cual solo contiene el método `actionPerformed()`.
  - Ver `TimeGoesByTest.java`