Interfaces

Contenido

Interfaces - parte 1

- ¿Qué es una interface?
- Usos y generalidades.
- Declaración.
- Implementación.
- Herencia de interfaces.
- Ventajas.

Interfaces - parte 2

- Interfaces funcionales.
- Interfaces funcionales predefinidas.
- Comparator<T> y Comparable<T>.
- Iterator<T> e Iterable<T>.

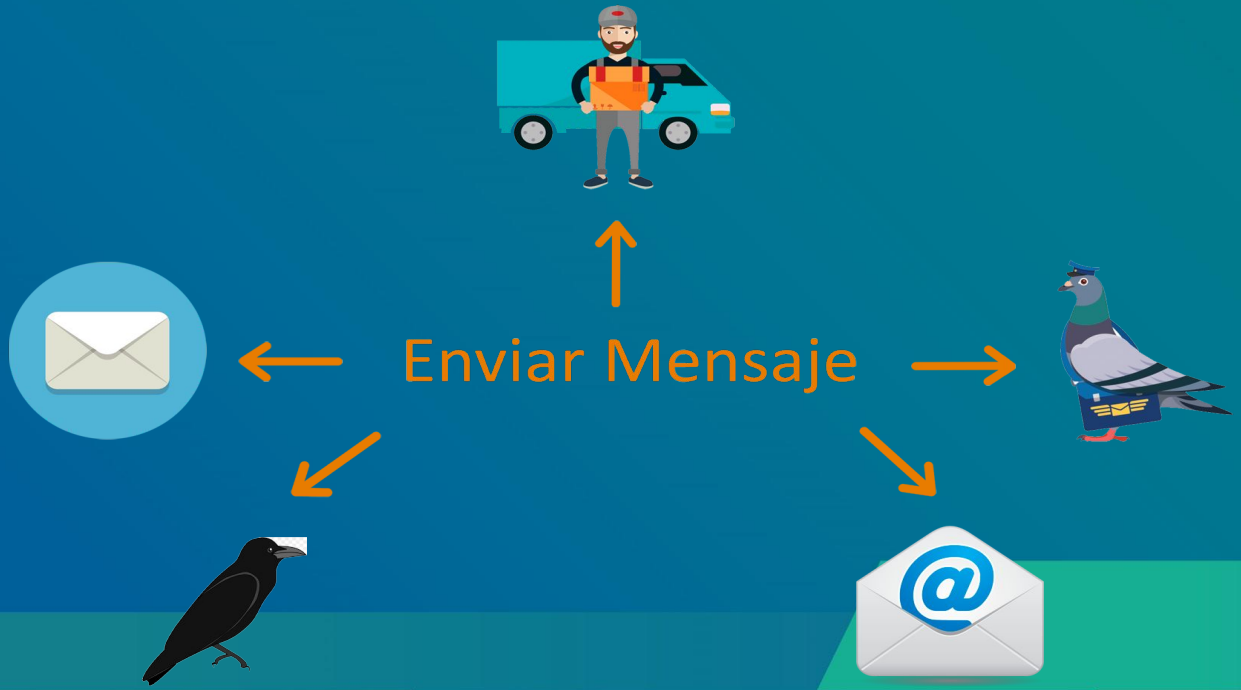
¿Qué es una interface?

Una interfaz es un medio común que nos permite agrupar **funcionalidades** que una clase luego **deberá** implementar.

Se puede decir que las interfaces establecen un **contrato** en el cual las clases que implementan la interfaz están **obligadas** a implementar sus funcionalidades.

Como las interfaces permiten tener las funcionalidades separadas de la implementación, se pueden tener distintas implementaciones en diferentes clases sin una relación fuerte, pero con la misma funcionalidad.

¿Qué es una interface?



Usos y generalidades

Permiten definir **métodos** y variables estáticas, finales y públicas, que en efecto son **constantes**.

No permiten definir ni **atributos** ni **constructores**.

Una clase puede implementar varias interfaces.

Todos los miembros definidos en una interfaz son, por definición, **públicos** y **abstractos** (*).

(*) Hasta JAVA 8.

Usos y generalidades

Desde JAVA 8, se implementaron los *métodos por defecto* y *métodos estáticos*.

Métodos por defecto: una interfaz puede tener métodos con una implementación predeterminada utilizando la palabra clave **default**.
Estos métodos proporcionan una implementación por defecto, pero las clases que implementan la interfaz ***pueden*** sobreescribir estos métodos.

Métodos estáticos: También se introdujeron métodos estáticos que pueden ser invocados directamente desde la interfaz, como si fueran de una clase normal.

Declaración

```java
public interface MiInterface{

    int CONSTANTE = 5;

    void metodo();

    default void metodoDefecto() {
        System.out.println("Método por defecto con implementación");
    }

    static void metodoEstatico() {
        System.out.println("Método estático");
    }
}
```

Implementación

```java
public class Clase implements MiInterface {

    @Override
    public void metodo() {
        System.out.println("Implemento el método. (OBLIGATORIAMENTE)");
    }

    @Override
    public void metodoDefecto() {
        MiInterface.super.metodoDefecto();
        System.out.println("Agrego implementación. (OPCIONAL)");
    }

    public static void MetodoEstatico(){
        MiInterface.metodoEstatico();
        System.out.println("Agrego implementación. (OPCIONAL)");
    }
}
```

Herencia de interfaces

Una interfaz puede heredar de otra interfaz, lo que permite extender la funcionalidad sin romper el principio de herencia múltiple.

```java
public interface OtraInterface extends MiInterface {

    void otroMetodo();
}
```

Una clase que implemente una interfaz derivada debe implementar todos los métodos de la jerarquía.

Ventajas en el uso de interfaces

Desacoplamiento: Las interfaces permiten crear sistemas menos acoplados, ya que una clase no necesita conocer la implementación específica de otra clase.

Herencia múltiple: Permite a una clase implementar múltiples interfaces y, por lo tanto, tener múltiples comportamientos, algo que no es posible con clases.

Polimorfismo: Proporciona una manera flexible de utilizar objetos de diferentes tipos de manera uniforme a través de un tipo común.

Ejercitación