Tema X: INTERFACES

**"Desde el punto de vista de un programador,
el usuario no es más que un periférico que teclea
cuando se le envía una petición de lectura"**

Protocolo de comunicación entre dos objetos no relacionados entre sí.

Una interfaz es un tipo especial de clase que define la especificación de un conjunto de métodos. El conjunto de estos métodos garantizan que cualquier clase que implementa una interfaz tiene que proporcionar esos métodos.

Declaración de una interface:

```
interface nombreI{
        declaración de métodos
}
```

Clase que implementa una interface:

```
class nombreclase implements nombreI{
        cuerpos de los métodos del interface,
        datos y métodos propios
        }
```

Propiedades:

- En un interface los métodos no se implementan, solo se declaran.
- Si un interface tiene alguna variable esta es implícitamente public, static y final.
- Si una clase implementa una interfaz debe codificar todos los métodos de la misma.
- No puedo instanciar objetos de un interfaz con new.
- Sin embargo puedo asignar un objeto de una clase que implementa un interface a un objeto variable declarado de la clase interface.

EjemploInterface1:

Tenemos un interface Machine, que tiene dos métodos abstractos

```
public interface Machine {
        public String suena();
        public void reset();

}
```

Por otro lado tenemos una clase Vehiculo que tiene sus datos y métodos, pero implementa el interface Machine, así que tengo que codificar los métodos de dicha interface en la clase Vehiculo:

```
public class Vehiculo implements Machine{
	private String matrícula;
	private int cuentaKms;
	public Vehiculo(String matrícula) {
		this.matrícula = matrícula;
		this.cuentaKms = 0;
	}

	public void recorrer(int kms){
		cuentaKms+=kms;
	}
	@Override
	public String suena() {
		// TODO Auto-generated method stub
		return "Piiiii";
	}
	@Override
	public void reset() {
		// TODO Auto-generated method stub
		cuentaKms=0;

	}

	@Override
	public String toString() {
		return "Vehiculo [matrícula=" + matrícula + ", cuentaKms=" +
cuentaKms + "]";
	}


}
```

Otras propiedades de las interfaces:

- Una clase puede implementar múltiples interfaces, se separan con coma.
- Se puede hacer que una interfaz B derive de otra interfaz A usando extends. En este caso, si una clase C implementa el interfaz B, tiene que codificar todos los métodos de A y de B.

Interfaces Útiles

Interface Comparable

La interface **java.lang.Comparable** puede ser implementada por cualquier clase cuyos objetos puedan ser ordenados.

Tiene un único método compareTo que devuelve un valor menor, igual o mayor que cero si el objeto actual es menor, igual o mayor que el objeto que se le pasa como parámetro.

```
public interface Comparable{
        int compareTo(Object o);
}
```

La clase Collections tiene un método sort static para ordenar los elementos de una lista. La condición es que los elementos de la lista deben implementar el interfaz Comparable, es decir, tienen que tener definido el método compareTo.

Ver EjemploInterface2.

Interface Comparator

Para ordenar objetos en base a distintos criterios en distintas situaciones, empleados ordenados alfabéticamente o por edades según nos interese. Se utiliza el interfaz **java.util.Comparator**.

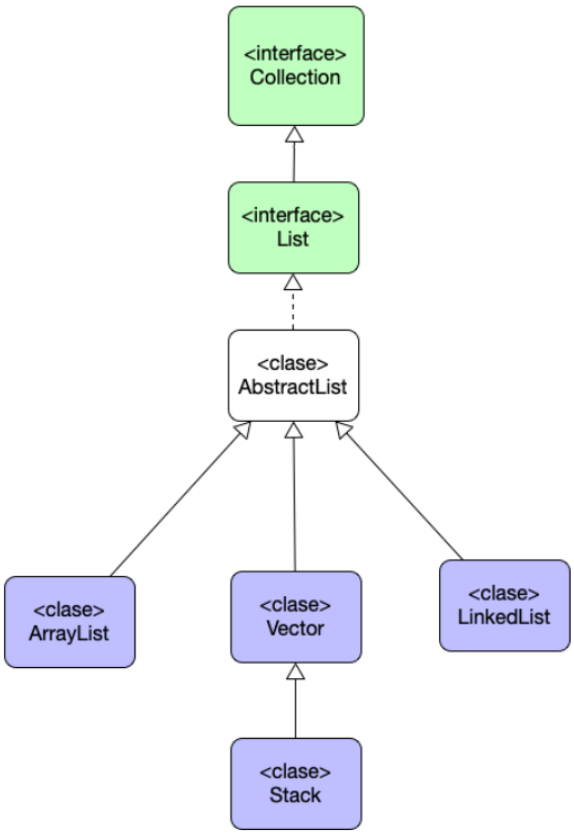
```
public interface Comparator{
        int compare(Object o1, Object o2);
        }
```

El método compare recibe los dos objetos a ordenar y el criterio de ordenación en cada situación.
Dicho criterio debe ser una clase previamente definida que implemente el interfaz Comparator. Estas clases pueden ser externas a la clase de los elementos que queremos ordenar o bien internas anidar(clases internas) tantas clases dentro de la clase de objetos a ordenar como criterios de ordenación tenga y cada una de estas clases implementar la interfaz dependiendo de dicho criterio.

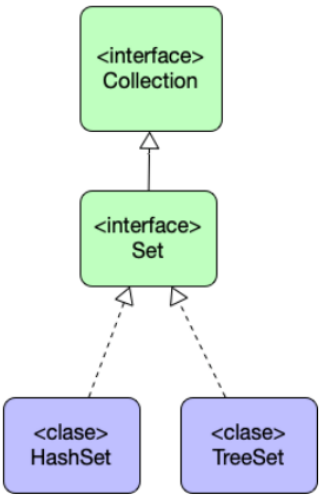
Ver EjemploInterface3.

Interface Collection

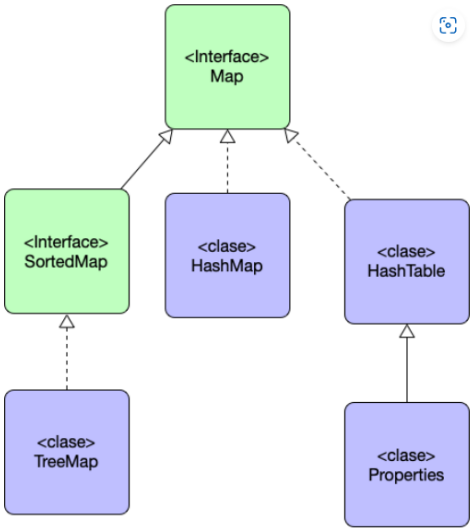
Collection es un Interface del que extiende otro, List. La clase AbstractList implementa el interface List, y de esa clase derivan ArrayList, Vector y LinkedList.



Del interface Collection también deriva el interface Set. Las clases HashSet y TreeSet implementan este interface.



La clase HashMap implementa el interface Map.



[Java Collections Framework y su estructura - Arquitectura Java]