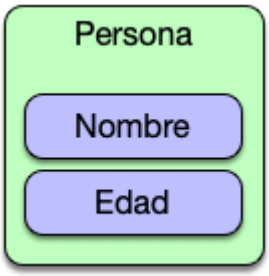
Tabla de Contenidos

El uso de Java Comparable es uno de los conceptos más fundamentales en cuando a manejo de las APIS básicas . Java soporta el interface Comparable para comparar objetos entre sí y poderlos ordenar. Vamos a ver un ejemplo sencillo de cómo realizar estas tareas a través de la clase Persona que contiene las propiedades de nombre y apellidos.



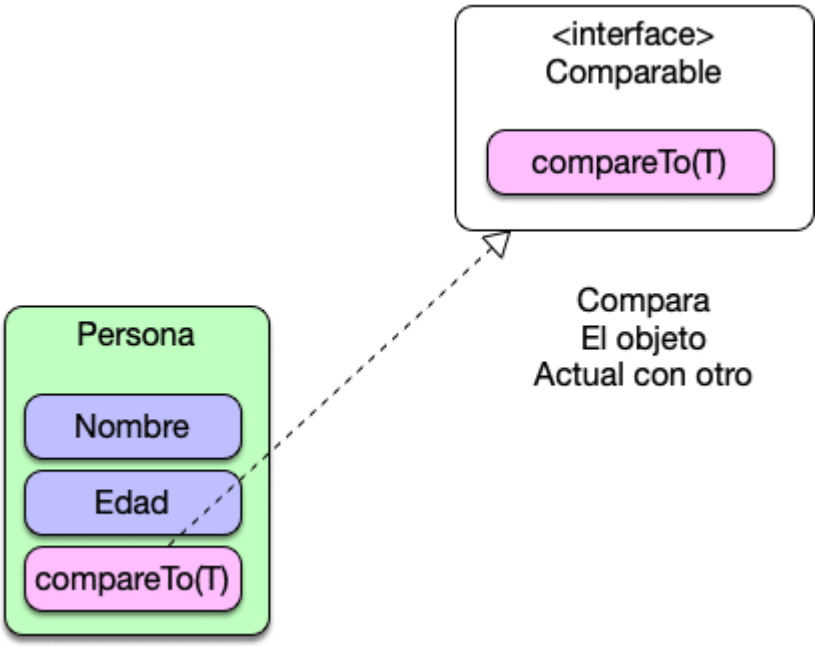

```
package com.arquitecturajava.ejemplo4;

public class Persona  implements Comparable<Persona>{

	private String nombre;
	private int edad;
	public String getNombre() {
		return nombre;
	}
	public void setNombre(String nombre) {
		this.nombre = nombre;
	}
	public int getEdad() {
		return edad;
```

```
	}
	public void setEdad(int edad) {
		this.edad = edad;
	}
	public Persona(String nombre, int edad) {
		super();
		this.nombre = nombre;
		this.edad = edad;
	}
	@Override
	public int compareTo(Persona o) {
		if (this.getEdad()>o.getEdad()) {
			return 1;
		}else if (this.getEdad()<o.getEdad()) {
			return -1;
		}else {
			return 0;
		}
	}
}
```

En este caso la Persona implementa el interface Comparable que contiene el método compareTo .



Java Comparable y su implementación

Este método es el usado para comparar dos objetos y decidir cuando uno es mayor que otro . Devuelve 1 si el objeto actual es mayor que el que pasamos como parámetro. Devuelve 0 si en la comparación se produce una relación de igualdad y devuelve -1 si es menor . De esta forma podemos apoyándonos en Java Comparable ordenar un ArrayList de objetos de forma rápida por ejemplo por la propiedad edad que es de lo que se encarga el código que hemos construido. Si construimos un programa main podremos ordenar una lista por edad .

```
package com.arquitecturajava.ejemplo4;

import java.util.ArrayList;
import java.util.Arrays;
import java.util.Collections;

public class Principal {

        public static void main(String[] args) {
                Persona p1= new Persona ("pedro",20);
```

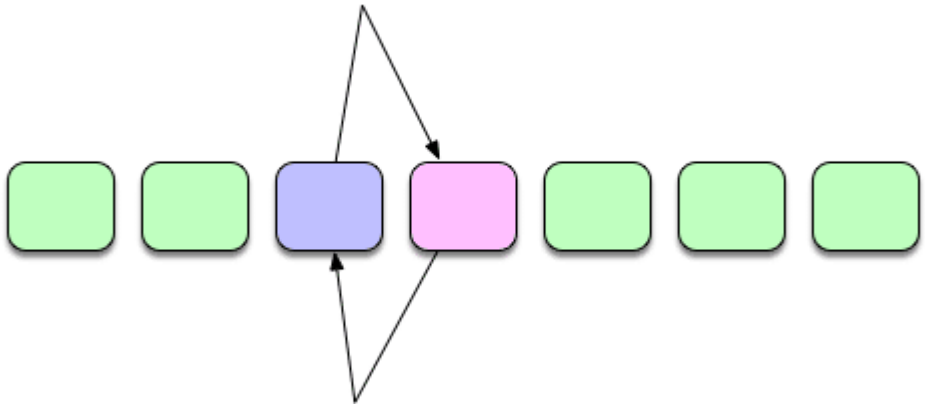
```
                Persona p2= new Persona ("juan",30);
                Persona p3= new Persona ("maria",40);
                Persona p4= new Persona ("gema",15);
                ArrayList<Persona> lista= new ArrayList<Persona>();
                lista.add(p1);
                lista.add(p2);
                lista.add(p3);
                lista.add(p4);
                Collections.sort(lista);
                for (Persona p :lista) {
                        System.out.print(p.getNombre());
                        System.out.println(p.getEdad());
                }

        }

}
```

De esta forma cuando imprimimos la lista el resultado aparecerá ordenado por edad.

Muchas veces me preguntan como Java ordena la lista , en este caso se apoya en el interface y va comprobando un elemento con el siguiente intercalando las posiciones si es necesario:



Otras ordenaciones

Para abordar otro tipo de ordenaciones necesitaremos hacer uso del interface Comparator

Otros artículos relacionados

- Java ArrayList for y sus opciones
- Java ArrayList count y las APIs de Java.
- Java Collections List vs Set (I)