Difference between Comparable and Comparator

Comparable and Comparator both are interfaces and can be used to sort collection elements.

However, there are many differences between Comparable and Comparator interfaces that are given below.

|  |  |
| --- | --- |
| **Comparable** | **Comparator** |
| 1) Comparable provides a **single sorting sequence**. In other words, we can sort the collection on the basis of a single element such as id, name, and price. | The Comparator provides **multiple sorting sequences**. In other words, we can sort the collection on the basis of multiple elements such as id, name, and price etc. |
| 2) Comparable **affects the original class**, i.e., the actual class is modified. | Comparator **doesn't affect the original class**, i.e., the actual class is not modified. |
| 3) Comparable provides **compareTo() method** to sort elements. | Comparator provides **compare() method** to sort elements. |
| 4) Comparable is present in **java.lang** package. | A Comparator is present in the **java.util** package. |
| 5) We can sort the list elements of Comparable type by **Collections.sort(List)** method. | We can sort the list elements of Comparator type by **Collections.sort(List, Comparator)** method. |

Java Comparable Example

Let's see the example of a Comparable interface that sorts the list elements on the basis of age.

1. //Java Program to demonstrate the use of Java Comparable.
2. //Creating a class which implements Comparable Interface
3. **import** java.util.\*;
4. **import** java.io.\*;
5. **class** Student **implements** Comparable<Student>{
6. **int** rollno;
7. String name;
8. **int** age;
9. Student(**int** rollno,String name,**int** age){
10. **this**.rollno=rollno;
11. **this**.name=name;
12. **this**.age=age;
13. }
14. **public** **int** compareTo(Student st){
15. **if**(age==st.age)
16. **return** 0;
17. **else** **if**(age>st.age)
18. **return** 1;
19. **else**
20. **return** -1;
21. }
22. }
23. //Creating a test class to sort the elements
24. **public** **class** TestSort3{
25. **public** **static** **void** main(String args[]){
26. ArrayList<Student> al=**new** ArrayList<Student>();
27. al.add(**new** Student(101,"Vijay",23));
28. al.add(**new** Student(106,"Ajay",27));
29. al.add(**new** Student(105,"Jai",21));
31. Collections.sort(al);
32. **for**(Student st:al){
33. System.out.println(st.rollno+" "+st.name+" "+st.age);
34. }
35. }

}

1. Compaarator
2. **class** Student{
3. **int** rollno;
4. String name;
5. **int** age;
6. Student(**int** rollno,String name,**int** age){
7. **this**.rollno=rollno;
8. **this**.name=name;
9. **this**.age=age;
10. }
11. }

**AgeComparator.java**

1. **import** java.util.\*;
2. **class** AgeComparator **implements** Comparator<Student>{
3. **public** **int** compare(Student s1,Student s2){
4. **if**(s1.age==s2.age)
5. **return** 0;
6. **else** **if**(s1.age>s2.age)
7. **return** 1;
8. **else**
9. **return** -1;
10. }
11. }

**NameComparator.java**

This class provides comparison logic based on the name. In such case, we are using the compareTo() method of String class, which internally provides the comparison logic.

1. **import** java.util.\*;
2. **class** NameComparator **implements** Comparator<Student>{
3. **public** **int** compare(Student s1,Student s2){
4. **return** s1.name.compareTo(s2.name);
5. }
6. }

In this class, we are printing the values of the object by sorting on the basis of name and age.

1. //Java Program to demonstrate the use of Java Comparator
2. **import** java.util.\*;
3. **import** java.io.\*;
4. **class** TestComparator{
5. **public** **static** **void** main(String args[]){
6. //Creating a list of students
7. ArrayList<Student> al=**new** ArrayList<Student>();
8. al.add(**new** Student(101,"Vijay",23));
9. al.add(**new** Student(106,"Ajay",27));
10. al.add(**new** Student(105,"Jai",21));
12. System.out.println("Sorting by Name");
13. //Using NameComparator to sort the elements
14. Collections.sort(al,**new** NameComparator());
15. //Traversing the elements of list
16. **for**(Student st: al){
17. System.out.println(st.rollno+" "+st.name+" "+st.age);
18. }
20. System.out.println("sorting by Age");
21. //Using AgeComparator to sort the elements
22. Collections.sort(al,**new** AgeComparator());
23. //Travering the list again
24. **for**(Student st: al){
25. System.out.println(st.rollno+" "+st.name+" "+st.age);
26. }
28. }
29. }