Comparator Interface in Java with Examples

Difficulty Level : Medium • Last Updated : 24 Mar, 2021

A comparator interface is used to order the objects of user-defined classes. A comparator object is capable of comparing two objects of two different classes. Following function compare obj1 with obj2

Syntax:


```
public int compare(Object obj1, Object obj2):
```

Suppose we have an Array/ArrayList of our own class type, containing fields like roll no, name, address, DOB, etc, and we need to sort the array based on Roll no or name?

Method 1: One obvious approach is to write our own sort() function using one of the standard algorithms. This solution requires rewriting the whole sorting code for different criteria like Roll No. and Name.

Method 2: Using comparator interface- Comparator interface is used to order the objects of a user-defined class. This interface is present in java.util package and contains 2 methods compare(Object obj1, Object obj2) and equals(Object element). Using a comparator, we can sort the elements based on data members. For instance, it may be on roll no, name, age, or anything else.

Method of Collections class for sorting List elements is used to sort the elements of List by the given comparator.

```
// To sort a given list. ComparatorClass must implement
// Comparator interface.
public void sort(List list, ComparatorClass c)
```

How does Collections.Sort() work?

Internally the Sort method does call Compare method of the classes it is sorting. To compare two elements, it asks “Which is greater?” Compare method returns -1, 0, or 1 to say if it is less than, equal, or greater to the other. It uses this result to then determine if they should be swapped for their sort.

Working Program:

Java

```
// Java program to demonstrate working of Comparator
// interface
import java.io.*;
import java.lang.*;
import java.util.*;

// A class to represent a student.
class Student {
    int rollno;
    String name, address;

    // Constructor
    public Student(int rollno, String name, String address)
    {
        this.rollno = rollno;
        this.name = name;
        this.address = address;
    }

    // Used to print student details in main()
    public String toString()
    {
        return this.rollno + " " + this.name + " "
            + this.address;
    }
}

class Sortbyroll implements Comparator<Student> {
    // Used for sorting in ascending order of
    // roll number
    public int compare(Student a, Student b)
    {
        return a.rollno - b.rollno;
    }
}

class Sortbyname implements Comparator<Student> {
    // Used for sorting in ascending order of
    // name
    public int compare(Student a, Student b)
    {
        return a.name.compareTo(b.name);
    }
```

```
}

// Driver class
class Main {
    public static void main(String[] args)
    {
        ArrayList<Student> ar = new ArrayList<Student>();
        ar.add(new Student(111, "bbbb", "london"));
        ar.add(new Student(131, "aaaa", "nyc"));
        ar.add(new Student(121, "cccc", "jaipur"));

        System.out.println("Unsorted");
        for (int i = 0; i < ar.size(); i++)
            System.out.println(ar.get(i));

        Collections.sort(ar, new Sortbyroll());

        System.out.println("\nSorted by rollno");
        for (int i = 0; i < ar.size(); i++)
            System.out.println(ar.get(i));

        Collections.sort(ar, new Sortbyname());

        System.out.println("\nSorted by name");
        for (int i = 0; i < ar.size(); i++)
            System.out.println(ar.get(i));
    }
}
```

Output

```
Unsorted
111 bbbb london
131 aaaa nyc
121 cccc jaipur

Sorted by rollno
111 bbbb london
121 cccc jaipur
131 aaaa nyc

Sorted by name
131 aaaa nyc
111 bbbb london
```

```
121 cccc jaipur
```

By changing the return value inside the compare method, you can sort in any order that you wish to. Eg. for descending order just change the positions of 'a' and 'b' in the above compare method.

Sort collection by more than one field:

In previous articles, we have discussed how to sort the list of objects on the basis of a single field using Comparable and Comparator interface But, what if we have a requirement to sort ArrayList objects in accordance with more than one fields like firstly, sort according to the student name and secondly, sort according to student age.
Below is the implementation of the above approach:

Java

```java
// Java program to demonstrate working of Comparator
// interface more than one field

import java.util.ArrayList;
import java.util.Collections;
import java.util.Comparator;
import java.util.Iterator;
import java.util.List;

class Student {

    // instance member variables
    String Name;
    int Age;

    // parameterized constructor
    public Student(String Name, Integer Age)
    {
        this.Name = Name;
        this.Age = Age;
    }

    public String getName() { return Name; }

    public void setName(String Name) { this.Name = Name; }
```

```
public Integer getAge() { return Age; }

public void setAge(Integer Age) { this.Age = Age; }

// overriding toString() method
@Override public String toString()
{
    return "Customer{"
        + "Name=" + Name + ", Age=" + Age + '}';
}

static class CustomerSortingComparator
    implements Comparator<Student> {

    @Override
    public int compare(Student customer1,
                       Student customer2)
    {

        // for comparison
        int NameCompare = customer1.getName().compareTo(
            customer2.getName());
        int AgeCompare = customer1.getAge().compareTo(
            customer2.getAge());

        // 2-level comparison
        return (NameCompare == 0) ? AgeCompare
                                  : NameCompare;
    }
}

public static void main(String[] args)
{

    // create ArrayList to store Student
    List<Student> al = new ArrayList<>();

    // create customer objects using constructor
    // initialization
    Student obj1 = new Student("Ajay", 27);
    Student obj2 = new Student("Sneha", 23);
    Student obj3 = new Student("Simran", 37);
    Student obj4 = new Student("Ajay", 22);
    Student obj5 = new Student("Ajay", 29);
    Student obj6 = new Student("Sneha", 22);

    // add customer objects to ArrayList
    al.add(obj1);
```

```
        al.add(obj2);
        al.add(obj3);
        al.add(obj4);
        al.add(obj5);
        al.add(obj6);

        // before Sorting arraylist: iterate using Iterator
        Iterator<Student> custIterator = al.iterator();

        System.out.println("Before Sorting:\n");
        while (custIterator.hasNext()) {
            System.out.println(custIterator.next());
        }

        // sorting using Collections.sort(al, comparator);
        Collections.sort(al,
                         new CustomerSortingComparator());

        // after Sorting arraylist: iterate using enhanced
        // for-loop
        System.out.println("\n\nAfter Sorting:\n");
        for (Student customer : al) {
            System.out.println(customer);
        }
    }
}
```

Output

```
Before Sorting:

Customer{Name=Ajay, Age=27}
Customer{Name=Sneha, Age=23}
Customer{Name=Simran, Age=37}
Customer{Name=Ajay, Age=22}
Customer{Name=Ajay, Age=29}
Customer{Name=Sneha, Age=22}


After Sorting:

Customer{Name=Ajay, Age=22}
```

```
Customer{Name=Ajay, Age=27}
Customer{Name=Ajay, Age=29}
Customer{Name=Simran, Age=37}
Customer{Name=Sneha, Age=22}
Customer{Name=Sneha, Age=23}
```

This article is contributed by **Rishabh Mahrsee**. If you like GeeksforGeeks and would like to contribute, you can also write an article and mail your article to contribute@geeksforgeeks.org. See your article appearing on the GeeksforGeeks main page and help other Geeks. Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

Previous
Functional Interfaces In Java

Next
List of all Java Keywords

Article Contributed By :

GeeksforGeeks

Vote for difficulty

Current difficulty : Medium

Easy | Normal | Medium | Hard | Expert

Improved By : Rajput-Ji, gfg_user, deepam, vaibhavchotani001, amit98

Article Tags : Java

Practice Tags : Java

Report Issue

Which of the following dog food products have you heard of?

SELECT UP TO 5 ANSWERS

- [ ] Rachael Ray Nutrish Big Life
- [ ] Blue Buffalo Life Protection
- [ ] Purina ONE
- [ ] IAMS Adult Formula
- [ ] Nutro Adult Formula

Powered By Nielsen
View Privacy Policy

Vote For Results

WHAT'S NEW

DSA Live Classes for Working Professionals

View Details

Competitive Programming Live Classes for Students

View Details

Complete Interview Preparation Course

View Details

MOST POPULAR IN JAVA

Split() String method in Java with examples

Reverse a string in Java

Arrays.sort() in Java with examples

How to add an element to an Array in Java?

Initialize an ArrayList in Java

MORE RELATED ARTICLES IN JAVA

How to iterate any Map in Java

Different ways of Reading a text file in Java

Difference between == and .equals() method in Java

Singleton Class in Java

Initializing a List in Java

GeeksforGeeks

5th Floor, A-118,
Sector-136, Noida, Uttar Pradesh - 201305

feedback@geeksforgeeks.org