Java Comparator interface

Java Comparator interface is used to order the objects of user-defined class.

This interface is found in java.util package and contains 2 methods compare(Object obj1,Object obj2) and equals(Object element).

It provides multiple sorting sequence i.e. you can sort the elements on the basis of any data member, for example rollno, name, age or anything else.

compare() method

public int compare(Object obj1,Object obj2): compares the first object with second object.

Collections class

Collections class provides static methods for sorting the elements of collection. If collection elements are of Set or Map, we can use TreeSet or TreeMap. But we cannot sort the elements of List. Collections class provides methods for sorting the elements of List type elements also.

Method of Collections class for sorting List elements

public void sort(List list, Comparator c): is used to sort the elements of List by the given Comparator.

Java Comparator Example (Non-generic Old Style)

Let's see the example of sorting the elements of List on the basis of age and name. In this example, we have created 4 java classes:

1. Student.java

2. AgeComparator.java

3. NameComparator.java

4. Simple.java

Student.java

This class contains three fields rollno, name and age and a parameterized constructor.

```
class Student{
int rollno;
String name;
int age;
Student(int rollno,String name,int age){
this.rollno=rollno;
this.name=name;
this.age=age;
}
}
```

AgeComparator.java

This class defines comparison logic based on the age. If age of first object is greater than the second, we are returning positive value, it can be any one such as 1, 2 , 10 etc. If age of first object is less than the second object, we are returning negative value, it can be any negative value and if age of both objects are equal, we are returning 0.

```java
import java.util.*;
class AgeComparator implements Comparator{
public int compare(Object o1,Object o2){
Student s1=(Student)o1;
Student s2=(Student)o2;

if(s1.age==s2.age)
return 0;
else if(s1.age>s2.age)
return 1;
else
return -1;
}
}
```

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.

```java
import java.util.*;
class NameComparator implements Comparator{
public int compare(Object o1,Object o2){
Student s1=(Student)o1;
Student s2=(Student)o2;

return s1.name.compareTo(s2.name);
}
}
```

Simple.java

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

```java
import java.util.*;
import java.io.*;

class Simple{
public static void main(String args[]){

ArrayList al=new ArrayList();
al.add(new Student(101,"Vijay",23));
al.add(new Student(106,"Ajay",27));
al.add(new Student(105,"Jai",21));


System.out.println("Sorting by Name...");


Collections.sort(al,new NameComparator());
Iterator itr=al.iterator();
while(itr.hasNext()){
Student st=(Student)itr.next();
System.out.println(st.rollno+" "+st.name+" "+st.age);
}


System.out.println("sorting by age...");


Collections.sort(al,new AgeComparator());
Iterator itr2=al.iterator();
while(itr2.hasNext()){
Student st=(Student)itr2.next();
System.out.println(st.rollno+" "+st.name+" "+st.age);
}


}
}
```

```
        Sorting by Name...
        106 Ajay 27
        105 Jai 21
        101 Vijay 23

        Sorting by age...
        105 Jai 21
        101 Vijay 23
        106 Ajay 27
```

Java Comparator Example (Generic)

Student.java

```java
class Student{
int rollno;
String name;
int age;
Student(int rollno,String name,int age){
this.rollno=rollno;
this.name=name;
this.age=age;
}
}
```

AgeComparator.java

```java
import java.util.*;
class AgeComparator implements Comparator<Student>{
public int compare(Student s1,Student s2){
if(s1.age==s2.age)
return 0;
else if(s1.age>s2.age)
return 1;
else
return -1;
}
}
```

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.

```java
import java.util.*;
class NameComparator implements Comparator<Student>{
public int compare(Student s1,Student s2){
return s1.name.compareTo(s2.name);
}
}
```

Simple.java

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

```java
import java.util.*;
import java.io.*;
class Simple{
public static void main(String args[]){

ArrayList<Student> al=new ArrayList<Student>();
al.add(new Student(101,"Vijay",23));
al.add(new Student(106,"Ajay",27));
al.add(new Student(105,"Jai",21));

System.out.println("Sorting by Name...");

Collections.sort(al,new NameComparator());
for(Student st: al){
System.out.println(st.rollno+" "+st.name+" "+st.age);
}

System.out.println("sorting by age...");

Collections.sort(al,new AgeComparator());
for(Student st: al){
```

```
System.out.println(st.rollno+" "+st.name+" "+st.age);

}


}
}
```

```
Output:Sorting by Name...
      106 Ajay 27
      105 Jai 21
      101 Vijay 23

      Sorting by age...
      105 Jai 21
      101 Vijay 23
      106 Ajay 27
```

← prev   next →

Share this page

facebook | twitter | google plus | pinterest | tumbler

Latest 4 Tutorials

CouchDB   Docker

Rails   RichFaces