



@techwithvishalraj



# Java

## Comparable

### v/s

## Comparator



Follow

Share



# Comparable

- **Comparable** is an **interface**.
- It is used to order the objects(descending/ascending).
- This interface is found in java.lang package.
- It contains only one method named **compareTo(Object)**.
- It is used to compare the current object with the specified object. It returns-
  - positive integer, if the current object is **greater** than the specified object.
  - **negative** integer, if the current object is **less** than the specified object.
  - **zero**, if the current object is **equal** to the specified object.
- We can sort the elements of:
  - String objects
  - Wrapper class objects
  - User-defined class objects
- **Note:** For reversing the Order just replace 1 with -1 and -1 with 1, rest of code as it is.

# compareTo() method in Comparable

```
class Student implements Comparable<Student>{
    String name;
    Integer age;
    public Student(String name, Integer age){
        this.name=name;
        this.age=age;
    }
    public Student() { }

    public String toString() {
        return "Student{ name=" + name + ", age=" + age + "}";
    }

    public int compareTo(Student s) {
        if(this.age == s.age){
            return 0;
        } else if (this.age>s.age) {
            return 1;
        }else{
            return -1;
        }
    }
}

public class Demo {
    public static void main(String[] args) {
        List<Student> students=new ArrayList<>();
        students.add(new Student("baburao", 50));
        students.add(new Student("shyam", 26));
        students.add(new Student("raju", 25));

        Collections.sort(students);
        System.out.println(students);
    }
}
```

A comparable object is capable of comparing itself with another object.

Output ->  
[Student{ name=raju, age=25},  
Student{ name=shyam, age=26},  
Student{ name=baburao, age=50}]

# Comparator

- **Comparator** is an **interface**.
- It is used to order the objects(descending/ascending).
- This interface is found in java.util package.
- It contains 2 methods-
  - **compare**(Object obj1, Object obj2), It compares the first object with the second object.  
It returns-
    - **positive** integer, if the current object is **greater** than the specified object.
    - **negative** integer, if the current object is **less** than the specified object.
    - **zero**, if the current object is **equal** to the specified object.
  - **equals**(Object element), It is used to check whether specified objects are equal class reference or not irrespective of their value.
- We can sort the elements of:
  - String objects
  - Wrapper class objects
  - User-defined class objects

# compare() method in Comparator

```
class Student implements Comparator<Student>{
    String name;
    Integer age;
    public Student(String name, Integer age){
        this.name=name;
        this.age=age;
    }
    public Student() { }

    public String toString() {
        return "Student{ name=" + name + ", age=" + age + '}';
    }

    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;
        }
    }
}

public class Demo {
    public static void main(String[] args) {
        List<Student> students=new ArrayList<>();
        students.add(new Student("baburao", 50));
        students.add(new Student("shyam", 26));
        students.add(new Student("raju", 25));

        Student student=new Student();
        Collections.sort(students, student);
        System.out.println(students);
    }
}
```

Comparator is external to the element type we are comparing.

For reversing the Order just replace 1 with -1 and -1 with 1.

Output ->  
[Student{ name=raju, age=25},  
Student{ name=shyam, age=26},  
Student{ name=baburao, age=50}]

# equals() method in Comparator

```
class Student implements Comparator<Student>{
    String name;
    Integer age;

    public Student(String name, Integer age){
        this.name=name;
        this.age=age;
    }

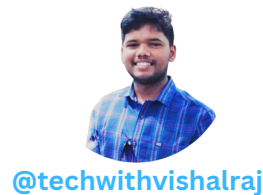
    public boolean equals(Object obj) {
        // Check if the obj object is also a Student.
        if (this == obj) {
            return true;
        }
        if (obj == null || getClass() != obj.getClass()) {
            return false;
        }
        return true;
    }
}

public class Demo {
    public static void main(String[] args) {
        Student student1=new Student("raju", 50);
        Student student2=new Student("shyam", 50);

        // Checking equality of two Comparator objects
        System.out.println(student1.equals(student2));
    }
}
```

Output -> true  
//not check the values.

# Comparator implemented Using Streams And Lambda



```
class Student {
    String name;
    Integer age;
    public Integer getAge() {
        return age;
    }
    public Student(String name, Integer age){
        this.name=name;
        this.age=age;
    }
    public String toString() {
        return "Student{ name=" + name + ", age=" + age + '}';
    }
}

public class Demo {
    public static void main(String[] args) {
        List<Student> students=new ArrayList<>();
        students.add(new Student("baburao", 50));
        students.add(new Student("shyam", 26));
        students.add(new Student("raju", 25));

        Collections.sort(students, (s1, s2)-> Integer.compare(s1.age, s2.age));
        Collections.sort(students, (s1, s2)-> {
            if(s1.age == s2.age){
                return 0;
            } else if (s1.age>s2.age) {
                return 1;
            }else{
                return -1;
            }
        });
        List<Student> sortedByAge = students.stream()
            .sorted(Comparator.comparingInt(Student::getAge)).toList();
    }
}
```

Output ->

```
[Student{ name=raju, age=25},
Student{ name=shyam, age=26},
Student{ name=baburao, age=50}]
```



@techwithvishalraj

*Thank  
you!*



vishal-bramhankar



techwithvishalraj



Vishall0317

