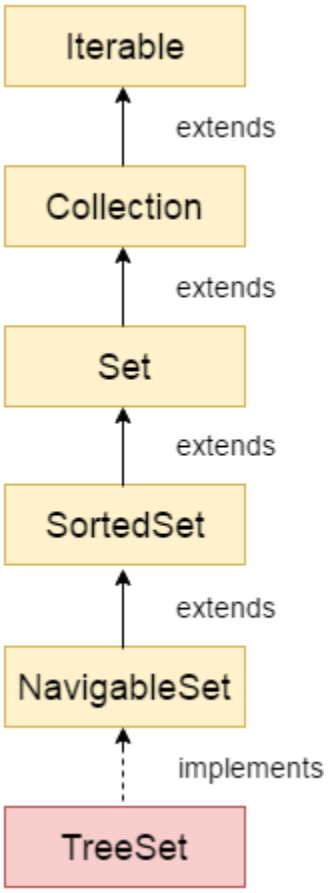
Java Tree Set class implements the Set interface that uses a tree for storage. It inherits Abstract Set class and implements the Navigable Set interface. The objects of the Tree Set class are stored in ascending order.



Features:

- Java Tree Set class contains unique elements only like HashSet.
- Java Tree Set class access and retrieval times are quite fast.
- Java Tree Set class doesn't allow null element.
- Java Tree Set class is non synchronized.
- Java Tree Set class maintains ascending order.

Syntax:

```
TreeSet<Integer> numbers = new TreeSet<>();
```

Example 1:

```
1. import java.util.*;
2. class TreeSet1{
3.  public static void main(String args[]){
4.   //Creating and adding elements
5.   TreeSet<String> al=new TreeSet<String>();
6.   al.add("Ravi");
7.   al.add("Vijay");
8.   al.add("Ravi");
9.   al.add("Ajay");
10.  //Traversing elements
11.  Iterator<String> itr=al.iterator();
12.  while(itr.hasNext()){
13.   System.out.println(itr.next());
14.  }
15. }
16.}
```

Output:

Ajay
Ravi
Vijay

Example 2:

```
1. import java.util.*;
2. class TreeSet3{
3.  public static void main(String args[]){
4.  TreeSet<Integer> set=new TreeSet<Integer>();
5.          set.add(24);
6.          set.add(66);
7.          set.add(12);
8.          set.add(15);
9.          System.out.println("Lowest Value: "+set.pollFirst());
10.         System.out.println("Highest Value: "+set.pollLast());
11. }
12.}
```

Output:

```
Lowest Value: 12
Highest Value: 66
```

Example 3:

```
import java.util.*;
public class TreeSetDemo {
  public static void main(String args[]) {
    // Create a tree set
    TreeSet ts = new TreeSet();
    // Add elements to the tree set
    ts.add("C");
    ts.add("A");
    ts.add("B");
    ts.add("E");
    ts.add("F");
    ts.add("D");
    System.out.println(ts);
  }
}
```

Output:

[A, B, C, D, E, F]

Why and when we use TreeSet ?

We prefer TreeSet in order to maintain the unique elements in the sorted order .

What is natural ordering in TreeSet ?

"Natural" ordering is the ordering implied by the implementation of Comparable interface by the objects in the TreeSet . Essentially RBTree must be able to tell which object is smaller than other object , and there are two ways to supply that logic to the RB Tree implementation :

1. We need to implement the Comparable interface in the class(es) used as objects in TreeSet.
2. Supply an implementation of the Comparator would do comparing outside the class itself.

How to convert HashSet to TreeSet object ?

One-liner : Set treeObject = new TreeSet(hashSetObject);

What is the difference between TreeSet and TreeMap ?

Ans. TreeSet contains only values as elements whereas TreeMap contains Key value pairs.