ECAP615

Programming in Java

Harjinder Kaur

Assistant Professor

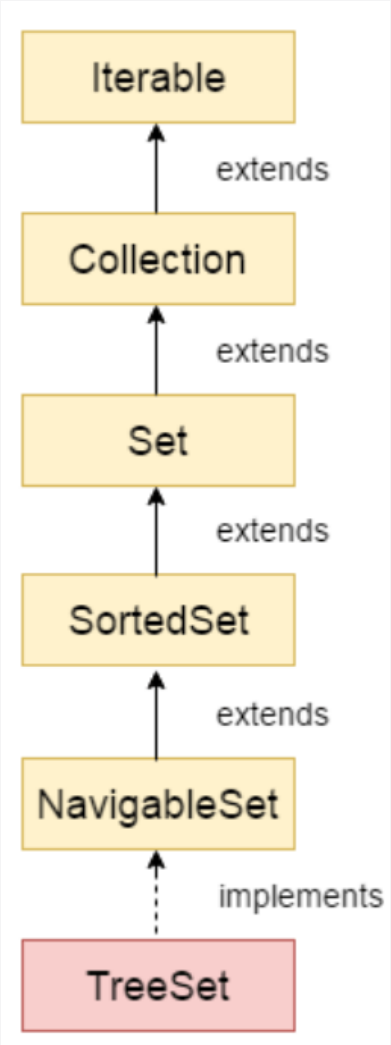
Learning Outcomes

After this lecture, you will be able to

- learn the basic concept of treeset Class
- understand the syntax of creating treeset class
- analyze the various methods available in treeset Class and their implementation

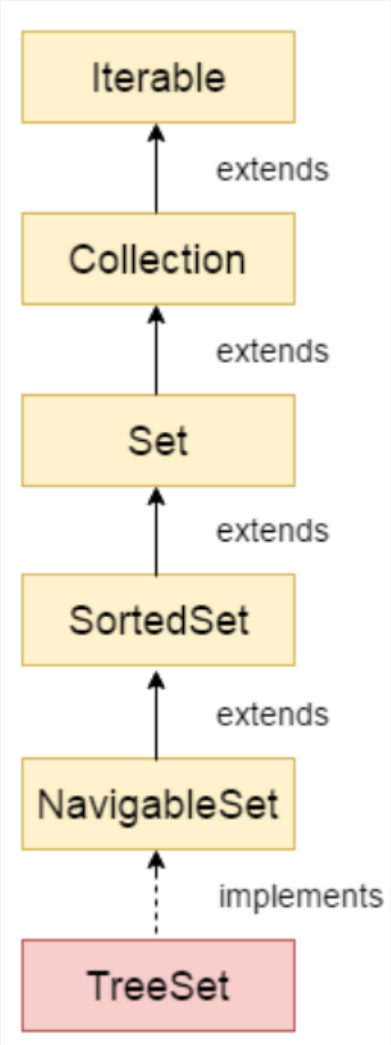
TreeSet class

- The TreeSet class of the Java collections framework provides the functionality of a tree data structure.
- Java TreeSet class implements the Set interface that uses a tree for storage.
- It inherits AbstractSet class and implements the NavigableSet interface.
- The objects of the TreeSet class are stored in ascending order.



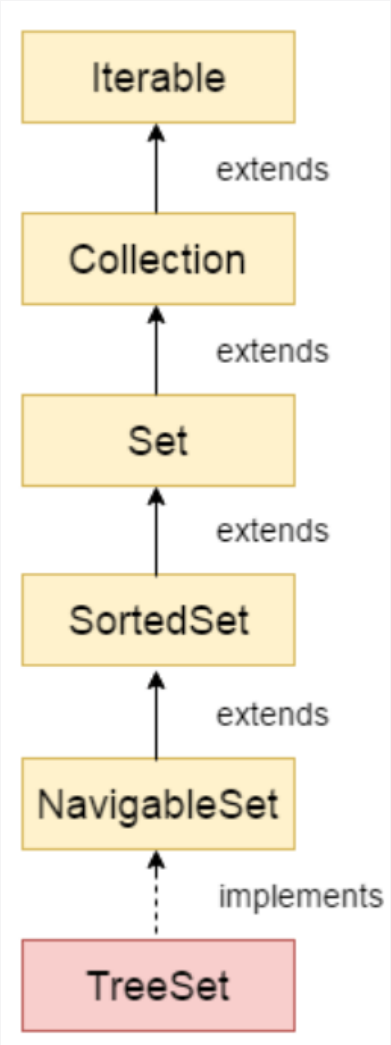
TreeSet class

- The TreeSet class of the Java collections framework provides the functionality of a tree data structure.
- Java TreeSet class implements the Set interface that uses a tree for storage.
- It inherits AbstractSet class and implements the NavigableSet interface.
- The objects of the TreeSet class are stored in ascending order.



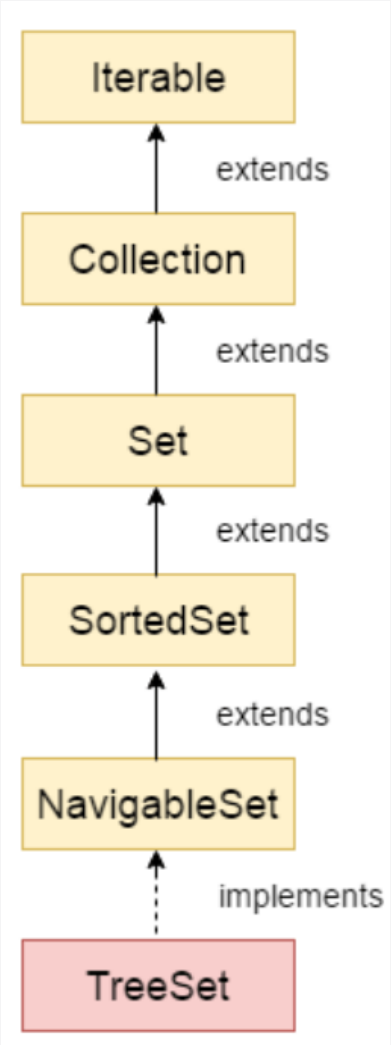
TreeSet class

- The TreeSet class of the Java collections framework provides the functionality of a tree data structure.
- Java TreeSet class implements the Set interface that uses a tree for storage.
- It inherits AbstractSet class and implements the NavigableSet interface.
- The objects of the TreeSet class are stored in ascending order.



TreeSet class

- The TreeSet class of the Java collections framework provides the functionality of a tree data structure.
- Java TreeSet class implements the Set interface that uses a tree for storage.
- It inherits AbstractSet class and implements the NavigableSet interface.
- **The objects of the TreeSet class are stored in ascending order.**



Points to remember

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

Points to remember

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

Points to remember

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

Points to remember

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

Points to remember

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

Creating a TreeSet

- In order to create a tree, set, we must import the java.util.TreeSet package first.
- Once we import the package, here is how we can create a TreeSet in Java.

Creating a TreeSet

- In order to create a tree, set, we must import the java.util.TreeSet package first.
- Once we import the package, here is how we can create a TreeSet in Java.

Syntax:

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

- Here, we have created a TreeSet without any arguments.

Methods of TreeSet

Insert Elements to TreeSet

- add() - inserts the specified element to the set
- addAll() - inserts all the elements of the specified collection to the set

Methods of TreeSet

Access TreeSet Elements

- To access the elements of a tree set, we can use the iterator() method.
- In order to use this method, we must import java.util.Iterator package.

Methods of TreeSet

Remove Elements

- remove() - removes the specified element from the set
- removeAll() - removes all the elements from the set

Methods for Navigation

1. first() and last() Methods

- first() - returns the first element of the set
- last() - returns the last element of the set

Methods for Navigation

2. pollfirst() and pollLast() Methods

- pollFirst() - returns and removes the first element from the set
- pollLast() - returns and removes the last element from the set

Methods for Navigation

3. ceiling(), floor(), higher() and lower() Methods

- higher(element) - Returns the lowest element among those elements that are greater than the specified element.

Methods for Navigation

3. ceiling(), floor(), higher() and lower() Methods

- lower(element) - Returns the greatest element among those elements that are less than the specified element.

Methods for Navigation

3. ceiling(), floor(), higher() and lower() Methods

- ceiling(element) - Returns the lowest element among those elements that are greater than the specified element. If the element passed exists in a tree set, it returns the element passed as an argument.

Methods for Navigation

3. ceiling(), floor(), higher() and lower() Methods

- floor(element) - Returns the greatest element among those elements that are less than the specified element. If the element passed exists in a tree set, it returns the element passed as an argument.

Methods for Navigation

4. headSet(), tailSet() and subSet() Methods

- The headSet() method returns all the elements of a tree set before the specified element.
- The tailSet() method returns all the elements of a tree set after the specified element (which is passed as a parameter) including the specified element.

Methods for Navigation

4. headSet(), tailSet() and subSet() Methods

- subSet(e1, bv1, e2, bv2)
- The subSet() method returns all the elements between e1 and e2 including e1.
- The bv1 and bv2 are optional parameters. The default value of bv1 is true, and the default value of bv2 is false.


That’s all for now…