SortedSet interface

04 April 2025 12:34

Sorted set is a child interface of Set interface. If we want to represent a group of individual objects, according to some sorting order, without duplicates, then we should go for SortedSet interface.

first() ⇒ 100
last() ⇒ 120
headSet(106) ⇒ [100, 101, 104]
tailSet(106) ⇒ [110, 115, 120]
subSet(101, 115) ⇒ [101, 104, 106, 110]
start ↑ included
end ↑ excluded

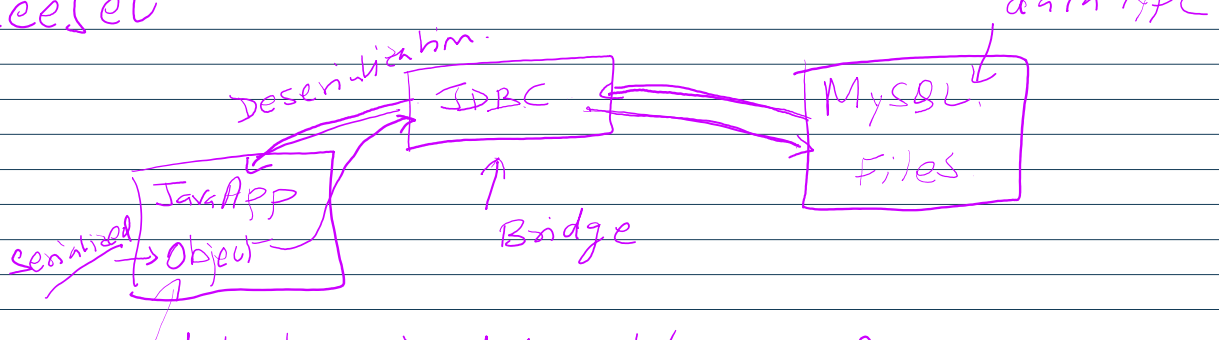
100
101
104
106
110
115
120

Comparator.comparator() ← returns comparator object describes underlying sorting technique. If we are using default natural sorting order, then we will get null.

Note: Default natural sorting order (DNSO) → Ascending order.
For Numbers → Ascending order
Strings → Alphabetical order

Set(I)
↓
SortedSet(I)
↓
→ NavigableSet(I) (1.6 V)
↓
TreeSet ← it implements Serializable, Cloneable but not RandomAccess.

TreeSet



serialize → Object

data type is different from MySQL.

TreeSet

The underlying data structure is balanced tree duplicate objects are not allowed. Insertion order is not preserved. heterogeneous objects are not allowed, otherwise we will get runtime exception saying ClassCastException. 'null' insertion is not possible.
TreeSet implements Serializable, Cloneable but not RandomAccess.
All objects will be inserted based on some sorting order. It may be default natural sorting order or customized sorting order.

Constructors:

TreeSet t = new TreeSet(); => Creates empty TreeSet object, where, although object entered into default natural sorting order.

TreeSet t = new TreeSet(Comparator c); => Creates an empty TreeSet object where the element will be inserted according to customized sorting order specified by the Comparator object.

TreeSet t = new TreeSet(SortedSet s); => Sorted order contained by object s.

TreeSet t = new TreeSet(Collection c); => Default natural sorting order It is an interconversion constructor. Yeah, of course. Inter conversion constructor.

Examples:

```
package Chapter1;
import java.util.TreeSet;
public class TreeSetDemo {
    public static void main(String[] args) {
        TreeSet t = new TreeSet<>();
        t.add("A");
        t.add("a");
        t.add("L");
        t.add("Z");
        t.add("B");
        t.add("C");
        System.out.println(t);
    }
}
```

Output:

```
[A, B, C, L, Z, a]
```

```
package Chapter1;
import java.util.TreeSet;
public class TreeSetDemoUsingStringBuffer {
    public static void main(String[] args) {
        TreeSet t = new TreeSet<>();
        t.add(new StringBuffer("A"));
        t.add(new StringBuffer("a"));
        t.add(new StringBuffer("C"));
        System.out.println(t);
```

```
    }
}
```

Output:

```
[A, C, a]
```

```
package Chapter1;
import java.util.TreeSet;
public class TreeSetDemoUsingStringBuffer {
    public static void main(String[] args) {
        TreeSet t = new TreeSet<>();
        t.add(new StringBuffer("A"));
        t.add(new StringBuffer("a"));
        t.add(new StringBuffer("C"));
        t.add("f");
        System.out.println(t);
    }
}
```

Homogeneous ✓
Comparable ✓
Serializable ✓

Output:

```
Exception in thread "main" java.lang.ClassCastException: class java.lang.StringBuffer cannot be cast to class java.lang.String (
java.lang.StringBuffer and java.lang.String are in module java.base of loader 'bootstrap')
        at java.base/java.lang.String.compareTo(String.java:141)
        at java.base/java.util.TreeMap.put(TreeMap.java:814)
        at java.base/java.util.TreeMap.put(TreeMap.java:534)
        at java.base/java.util.TreeSet.add(TreeSet.java:255)
        at Chapter1.TreeSetDemoUsingStringBuffer.main(TreeSetDemoUsingStringBuffer.java:12)
PS C:\Users\bluej\Documents\Students\RaghavISC\Java4Sem\Coding>
```