JAVA.UTIL.TREEMAP.LOWERKEY METHOD

Description

The **lowerKey***Kkey* method is used to return the greatest key strictly less than the given key, or null if there is no such key.

Declaration

Following is the declaration for **java.util.TreeMap.lowerKey** method.

```
public K lowerKey(K key)
```

Parameters

- **key** -- This is the key to be checked.

Return Value

The method call returns the greatest key less than key, or null if there is no such key.

Exception

- **ClassCastException** -- This exception is thrown if the specified key cannot be compared with the keys currently in the map.
- **NullPointerException** -- This exception is thrown if the specified key is null and this map uses natural ordering, or its comparator does not permit null keys.

Example

The following example shows the usage of java.util.TreeMap.lowerKey

```
package com.tutorialspoint;

import java.util.*;

public class TreeMapDemo {
   public static void main(String[] args) {
   // creating tree map
   TreeMap<Integer, String> treemap = new TreeMap<Integer, String>();

   // populating tree map
   treemap.put(2, "two");
   treemap.put(1, "one");
   treemap.put(3, "three");
   treemap.put(6, "six");
   treemap.put(5, "five");

   // getting lower key
   System.out.println("Checking lower key");
   System.out.println("Value is: "+ treemap.lowerKey(5));
   }
}
```

Let us compile and run the above program, this will produce the following result.

```
Checking lower key
Value is: 3
```