JAVA.UTIL.TREEMAP.TAILMAP METHOD

Description

The **tailMap***K fromKey* method is used to return a view of the portion of this map whose keys are greater than or equal to fromKey. The returned map is backed by this map, so changes in the returned map are reflected in this map, and vice-versa.

Declaration

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

```
public SortedMap<K,V> tailMap(K fromKey)
```

Parameters

- **fromKey** -- This is the low endpoint *inclusive* of the keys in the returned map.

Return Value

The method call returns a view of the portion of this map whose keys are greater than or equal to fromKey.

Exception

- **ClassCastException** -- This exception is thrown if fromKey is not compatible with this map's comparator.

- **NullPointerException** -- This exception is thrown if fromKey is null and this map uses natural ordering, or its comparator does not permit null keys.

- **IllegalArgumentException** -- This exception is thrown if this map itself has a restricted range, and fromKey lies outside the bounds of the range.

Example

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

```java
package com.tutorialspoint;

import java.util.*;

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

   System.out.println("Getting tail map");
   treemapincl=treemap.tailMap(3);
   System.out.println("Tail map values: "+treemapincl);
   }
}
```

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

```
Getting tail map
Tail map values: {3=three, 5=five, 6=six}
```