跟我学Java编程—应用自动排序的TreeMap

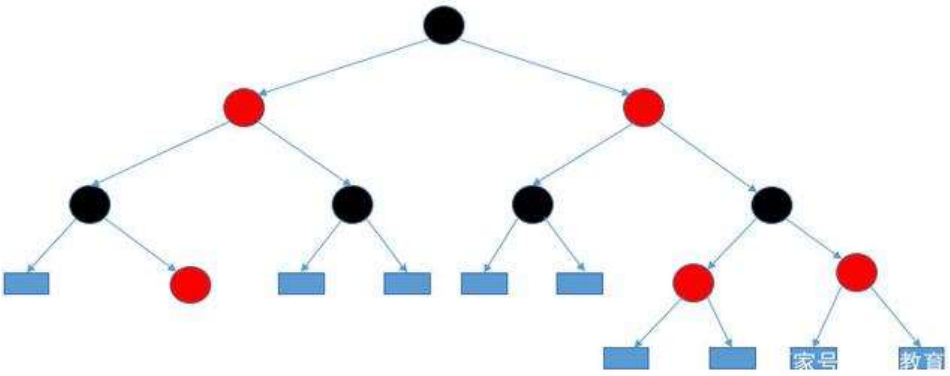
米粒教育
发布时间：18-04-27 20:16

前面介绍了Map接口的实现类LinkedHashMap，LinkedHashMap存储的元素是有序的，可以保持元素的插入顺序，但不能对元素进行自动排序。在一些编程应用场景中，如果在数据的存储过程中，能够自动对数据进行排序，将会极大提高编程效率，程序员无需再为数据排序编写必要的代码。例如，一般大量的数据都被存储在大型数据库中，程序员需要能够按照多个键对索引排序以提供搜索效率。

Map接口有一个重要的实现类TreeMap，TreeMap可以实现存储元素的自动排序。在TreeMap中，键值对之间按键有序，TreeMap的实现基础是平衡二叉树。

1、TreeMap 的存储结构

TreeMap使用的存储结构是平衡二叉树，也称为红黑树。它首先是一棵二叉树，具有二叉树所有的特性，即树中的任何节点的值大于它的左子节点，且小于它的右子节点，如果是一棵左右完全均衡的二叉树，元素的查找效率将获得极大提高。最坏的情况就是一边倒，只有左子树或只有右子树，这样势必会导致二叉树的检索效率大大降低。为了维持二叉树的平衡，程序员们提出了各种实现的算法，其中平衡二叉树就是其中的一种算法。平衡二叉树的数据结构如下图所示：



TreeMap存储结构

2、TreeMap 的构造函数

TreeMap 提供了三个常用的构造函数，说明如下：

- **TreeMap()**

使用该构造函数，TreeMap中的key按照自然排序进行排列。

- **TreeMap(Map<? extends K, ? extends V> copyFrom)**

作者最新文章

机器学习数学知识——矩阵加减运算及Python实现

使用Python如何开发用户界面程序？

机器学习数学知识及实现——什么是矩阵？

相关文章

java序列化系列之protobuf

用大白话告诉你：Java 后端到底是在做什么？

Java常用框架有哪些？先学哪一个比较有优势？

java虚拟机运行时数据区

● TreeMap(Comparator<? super K> comparator)

使用该构造函数，指定元素排序所用的比较器，key排列顺序由比较器指定。

2、TreeMap 元素的存取

同HashMap一样，TreeMap提供了get和put方法用于元素的存取。

● put(K key, V value)

该方法用于添加一个Entry（结点，包括key和value）到TreeMap对象，key为与指定值将要关联的键，value为使用指定键关联的值。如果key对应的值已经存在，则将key对应的值修改为value。

● V get(Object key)

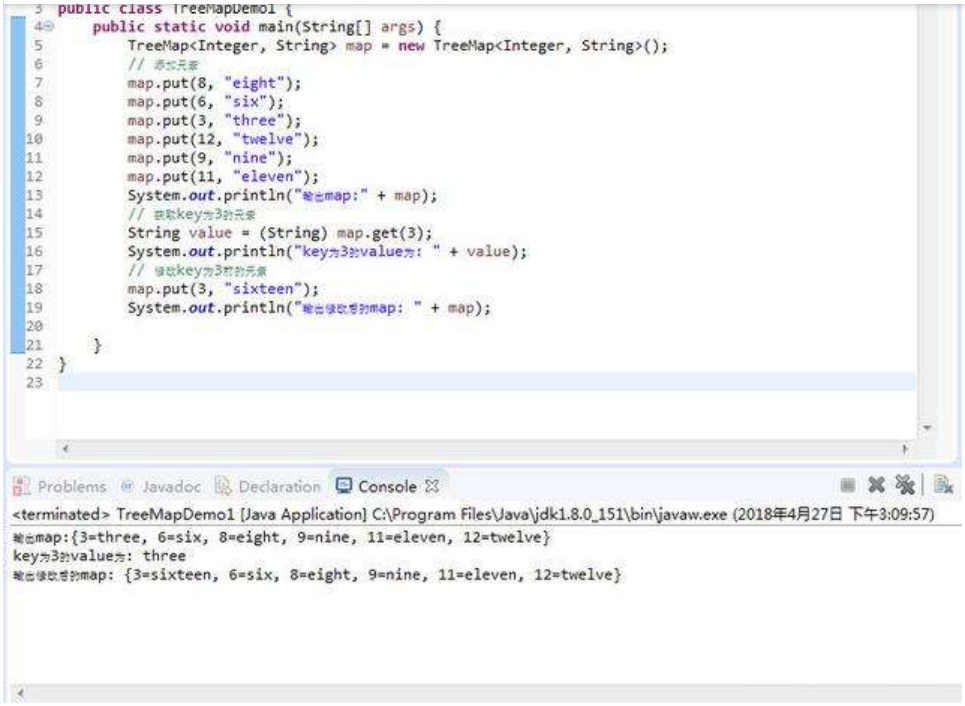
该方法用于获取指定key的value，key为与value相关联的键。

TreeMap 元素存取示例代码如下：

```
package com.milihua.treemapdemo;
import java.util.TreeMap;
public class TreeMapDemo1 {
    public static void main(String[] args) {
        TreeMap<Integer, String> map = new TreeMap<Integer, String>();
        // 添加元素
        map.put(8, "eight");
        map.put(6, "six");
        map.put(3, "three");
        map.put(12, "twelve");
        map.put(9, "nine");
        map.put(11, "eleven");
        System.out.println("输出map:" + map);
        // 获取key为3的元素
        String value = (String) map.get(3);
        System.out.println("key为3的value为: " + value);
        // 修改key为3前的元素
        map.put(3, "sixteen");
        System.out.println("输出修改后的map: " + map);

    }
}
```

该程序声明了TreeMap对象。首先用put方法添加了6个Entry结点，然后用get方法获取指定key的值，再用put方法修改指定key的值。程序输出结果如下图所示：



TreeMapDemo1输出结果

从图中输出结果可以看出，TreeMap按照传入的key进行自动排序。

3、TreeMap 的遍历

同HashMap、LinkedHashMap相同，TreeMap也不能用迭代器、foreach等方法遍历。如果需要遍历TreeMap，可以通过TreeMap的keySet方法返回TreeMap中key值的集合进行遍历。

TreeMap遍历代码如下：

```
package com.milihua.treemapdemo;
import java.util.Iterator;
import java.util.Map;
import java.util.Map.Entry;
import java.util.Set;
import java.util.TreeMap;
public class TreeMapDemo1 {
    public static void main(String[] args) {
        TreeMap<Integer, String> map = new TreeMap<Integer, String>();
        // 添加元素
        map.put(8, "eight");
        map.put(6, "six");
        map.put(3, "three");
        map.put(12, "twelve");
        map.put(9, "nine");
        map.put(11, "eleven");
        System.out.println("输出map:" + map);
        // 获取key为3的元素
        String value = (String) map.get(3);
        System.out.println("key为3的value为: " + value);
        // 修改key为3前的元素
        map.put(3, "sixteen");
        System.out.println("输出修改后的map: " + map);

        //获取Map中的所有key与value的对应关系
        Set<Entry<Integer, String>> entrySet =map.entrySet();
        //遍历Set集合
        Iterator<Map.Entry<Integer, String>> it=entrySet.iterator();
        while(it.hasNext()){
            //得到每一对对应关系
            Map.Entry<Integer, String> entry = it.next();
            //通过每一对对应关系获取对应的key
            Integer key = entry.getKey();
            //通过每一对对应关系获取对应的value
            String strValue = entry.getValue();
            System.out.println(key+"="+value);
        }

    }
}
```

程序获取TreeMap所有的键值对(Entry)对象，并以Set集合形式返回。然后，通过遍历包含键值对(Entry)对象的Set集合，得到每一个键值对。程序输出结果如下图所示：

```
        map.put(3, "sixteen");
        System.out.println("输出修改后的map: " + map);

        //获取Map中的所有key与value的对应关系
        Set<Entry<Integer, String>> entrySet =map.entrySet();
        //返回Set集合
        Iterator<Map.Entry<Integer, String>> it=entrySet.iterator();
        while(it.hasNext()){
            //得到每一对对应关系
            Map.Entry<Integer, String> entry = it.next();
            //通过每一对对应关系获取对应的key
            Integer key = entry.getKey();
            //通过每一对对应关系获取对应的value
            String strValue = entry.getValue();
            System.out.println(key+"="+value);
        }

    }
}
```

Problems Javadoc Declaration Console

```
<terminated> TreeMapDemo1 [Java Application] C:\Program Files\Java\jdk1.8.0_151\bin\javaw.exe (2018年4月27日 下午3:32:34)
输出map:{3=three, 6=six, 8=eight, 9=nine, 11=eleven, 12=twelve}
key为3的value为: three
输出修改后的map: {3=sixteen, 6=six, 8=eight, 9=nine, 11=eleven, 12=twelve}
3=three
6=three
8=three
9=three
11=three
12=three
```

遍历TreeMap输出结果

■ 知识点拨

HashMap通过hashcode对其内容进行快速查找，而 TreeMap中所有的元素都保持着某种固定的顺序，如果你需要得到一个有序的结果你就应该使用TreeMap。HashMap通常比TreeMap效率要高一些，一个是哈希表，一个是二叉树，建议多使用HashMap，在需要排序的Map时候才用TreeMap。