链滴

HashMap 和 Hashtable

作者：acer

原文链接：https://ld246.com/article/1534990529845

来源网站：链滴

HashMap和Hashtable区别

- HashMap线程不安全，操作不是同步的，而Hashtable是线程安全，里面的操作时同步的（一些常方法都用了synchronized标识符，如put,get等方法）
- HashMap碰到哈希冲突时是将新的Node放在array[index].next，Hashtable发生冲突时是将新的ode放在array[index]，将old array[index]放在array[index].next

线程安全的验证

```
package thread;

import java.util.*;

/**
 * @author:ace
 * @date:2018-08-21
 */
public class ThreadDemo {
    enum MapType{
        //HashMap
        HM,
        //Hashtable
        HT
    }
    public static void main(String[] args) {
        for (int i=0;i<20;i++){
            test(MapType.HT);
        }
    }
    private static void test(MapType type){
        try {

            //List<Integer> list = new ArrayList<>();
            Map<Integer,Integer> map;
            if(type==MapType.HM)
                map=new HashMap<>();
            else if(type==MapType.HT)
                map=new Hashtable<>();
            else
                map=null;
            //Hashtable<Integer, Integer> hashtable = new Hashtable<>();
            for (int i = 0; i < 100; i++) {
                //list.add(i);
                map.put(i,i);
            }
            Thread thread1 = new Thread(new Runnable() {
                @Override
                public void run() {
                    try {
                        /*
                        while (list.size() > 0) {
                            System.out.println(list.get(0));
```

```
                    list.remove(0);
                    Thread.sleep(100);
                }*/
                for (int i = 0; i < 1000; i++) {
                    map.put(i,i);
                }
                System.out.println("thread1: "+ map.get(500));
            } catch (Exception ex) {
                ex.printStackTrace();
            }
        }
    });
    Thread thread2 = new Thread(new Runnable() {
        @Override
        public void run() {
            try {
                /*
                while (list.size() > 0) {
                    System.out.println(list.get(0));
                    list.remove(0);
                    Thread.sleep(100);
                }*/
                for (int i = 1000; i < 2000; i++) {
                    map.put(i,i);
                }
                System.out.println("thread2: "+map.get(1500));
            } catch (Exception ex) {
                ex.printStackTrace();
            }
        }
    });
    thread1.start();
    thread2.start();

  } catch (Exception ex) {
    ex.printStackTrace();
  }
 }

}
```

当map为HashMap时，会出现为null的情况，因为不同的线程操作HashMap时不同步。map为Hashable时都是正常添加正常取出的，但是Hashtable会影响性能。

支持同步的HashMap

推荐ConcurrentHashMap类来做同步处理