java 容器都有哪些？

Collection的框架图



Collection 和 Collections 有什么区别？

* java.util.Collection 是一个集合接口（集合类的一个顶级接口）。它提供了对集合对象进行基本操作的通用接口方法。Collection接口在Java 类库中有很多具体的实现。Collection接口的意义是为各种具体的集合提供了最大化的统一操作方式，其直接继承接口有List与Set。
* Collections则是集合类的一个工具类/帮助类，其中提供了一系列静态方法，用于对集合中元素进行排序、搜索以及线程安全等各种操作

HashMap 和 Hashtable 有什么区别？

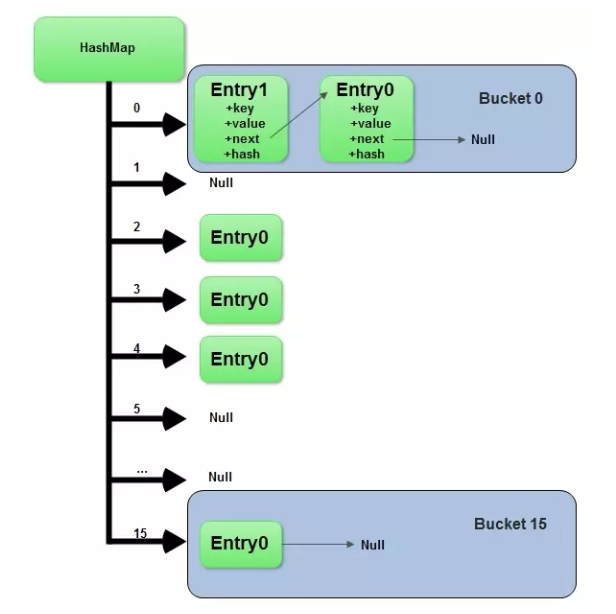
* hashMap去掉了HashTable 的contains方法，但是加上了containsValue（）和containsKey（）方法。
* hashTable同步的，而HashMap是非同步的，效率上比hashTable要高。
* hashMap允许空键值，而hashTable不允许。

如何决定使用 HashMap 还是 TreeMap？

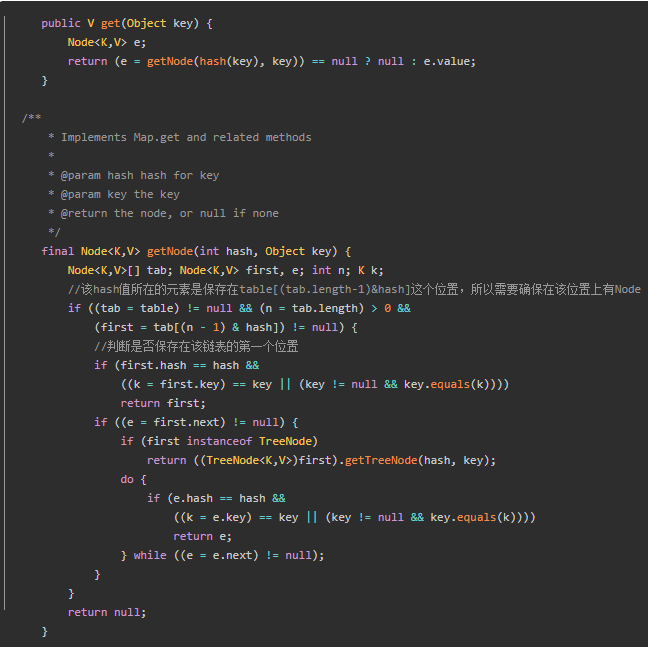
对于在Map中插入、删除和定位元素这类操作，HashMap是最好的选择。然而，TreeMap实现SortedMap接口，假如你需要对一个有序的key集合进行遍历，TreeMap是更好的选择。基于你的collection的大小，也许向HashMap中添加元素会更快，将map换为TreeMap进行有序key的遍历。

HashMap 的实现原理

HashMap是基于hashing的原理，我们使用put(key, value)存储对象到HashMap中，使用get(key)从HashMap中获取对象。当我们给put()方法传递键和值时，先对Key调用hashCode方法，来计算hash值，返回的hash值用来找bucket对象，来放entry键值



面我们来主要介绍下HashMap中最主要的两个方法：get(key)和put(key, value)



首先，如果key为null，则直接从哈希表的第一个位置table[0]对应的链表上查找。记住，key为null的键值对永远都放在以table[0]为头结点的链表中，当然不一定是存放在头结点table[0]中。如果key不为null，则先求的key的hash值，根据hash值找到在table中的索引，在该索引对应的单链表中查找是否有键值对的key与目标key相等，有就返回对应的value，没有则返回null

put(key, value)方法：

final V putVal(int hash, K key, V value, boolean onlyIfAbsent,  
 boolean evict) {  
 Node<K,V>[] tab; Node<K,V> p; int n, i;  
 if ((tab = table) == null || (n = tab.length) == 0)  
 n = (tab = resize()).length;  
 if ((p = tab[i = (n - 1) & hash]) == null)  
 tab[i] = newNode(hash, key, value, null);  
 else {  
 Node<K,V> e; K k;  
 if (p.hash == hash &&  
 ((k = p.key) == key || (key != null && key.equals(k))))  
 e = p;  
 else if (p instanceof TreeNode)  
 e = ((TreeNode<K,V>)p).putTreeVal(this, tab, hash, key, value);  
 else {  
 for (int binCount = 0; ; ++binCount) {  
 if ((e = p.next) == null) {  
 p.next = newNode(hash, key, value, null);  
 if (binCount >= *TREEIFY\_THRESHOLD* - 1) // -1 for 1st  
 treeifyBin(tab, hash);  
 break;  
 }  
 if (e.hash == hash &&  
 ((k = e.key) == key || (key != null && key.equals(k))))  
 break;  
 p = e;  
 }  
 }  
 if (e != null) { // existing mapping for key  
 V oldValue = e.value;  
 if (!onlyIfAbsent || oldValue == null)  
 e.value = value;  
 afterNodeAccess(e);  
 return oldValue;  
 }  
 }  
 ++modCount;  
 if (++size > threshold)  
 resize();  
 afterNodeInsertion(evict);  
 return null;  
}

如果key为null，则将其添加到table[0]对应的链表中，如果key不为null，则同样先求出key的hash值，根据hash值得出在table中的索引，而后遍历对应的单链表，如果单链表中存在与目标key相等的键值对，则将新的value覆盖旧的value，且将旧的value返回，如果找不到与目标key相等的键值对，或者该单链表为空，则将该键值对插入到单链表的头结点位置（每次新插入的节点都是放在头结点的位置）

当两个对象的hashcode相同会发生什么？

这就出现了所谓的hash冲突，解决hash冲突的方法有很多，HashMap底层是通过链表来解决hash冲突的。Jdk1.8以后当改链表长度大于8时采用红黑树来存储

负载因子(load factor)

默认的负载因子大小为0.75，也就是说，当一个map填满了75%的bucket时候，和其它集合类(如ArrayList等)一样，将会创建原来HashMap大小的两倍的bucket数组

重新调整HashMap大小存在什么问题?

当重新调整HashMap大小的时候，确实存在条件竞争，因为如果两个线程都发现HashMap需要重新调整大小了，它们会同时试着调整大小。在调整大小的过程中，存储在LinkedList中的元素的次序会反过来，因为移动到新的bucket位置的时候，HashMap并不会将元素放在LinkedList的尾部，而是放在头部，这是为了避免尾部遍历(tail traversing)。如果条件竞争发生了，那么就死循环了

为何HashMap的数组长度一定是2的次幂？

它通过h & (table.length -1)来得到该对象的保存位，而HashMap底层数组的长度总是2的n次方，这是HashMap在速度上的优化。当length总是2的n次方时，h& (length-1)运算等价于对length取模，也就是h%length，但是&比%具有更高的效率。

<https://blog.csdn.net/fjse51/article/details/53811465>

HashSet 的实现原理

HashSet底层由HashMap实现

HashSet的值存放于HashMap的key上

HashMap的value统一为PRESENT

ArrayList 和 LinkedList 的区别是什么？

最明显的区别是 ArrrayList底层的数据结构是数组，支持随机访问，而 LinkedList 的底层数据结构是双向循环链表，不支持随机访问。使用下标访问一个元素，ArrayList 的时间复杂度是 O(1)，而 LinkedList 是 O(n)

如何实现数组和 List 之间的转换？

List转换成为数组：调用ArrayList的toArray方法。

数组转换成为List：调用Arrays的asList方法

Queue 中 poll()和 remove()有什么区别？

poll() 和 remove() 都是从队列中取出一个元素，但是 poll() 在获取元素失败的时候会返回空，但是 remove() 失败的时候会抛出异常

迭代器 Iterator 是什么？

迭代器是一种设计模式，它是一个对象，它可以遍历并选择序列中的对象，而开发人员不需要了解该序列的底层结构。迭代器通常被称为“轻量级”对象，因为创建它的代价小。迭代器可以在迭代过程中删除底层集合的元素，但是不可以直接调用集合的remove(Object obj)删除，可以通过迭代器的remove()方法删除

(1) 使用方法iterator()要求容器返回一个Iterator。第一次调用Iterator的next()方法时，它返回序列的第一个元素。注意：iterator()方法是java.lang.Iterable接口,被Collection继承。

(2) 使用next()获得序列中的下一个元素。

(3) 使用hasNext()检查序列中是否还有元素。

(4) 使用remove()将迭代器新返回的元素删除。

Iterator是Java迭代器最简单的实现，为List设计的ListIterator具有更多的功能，它可以从两个方向遍历List，也可以从List中插入和删除元素。

Iterator 和 ListIterator 有什么区别？

Iterator可用来遍历Set和List集合，但是ListIterator只能用来遍历List。

Iterator对集合只能是前向遍历，ListIterator既可以前向也可以后向。

ListIterator实现了Iterator接口，并包含其他的功能，比如：增加元素，替换元素，获取前一个和后一个元素的索引，等等。