# Collection

1. **List.** :元素是有序的，元素可以重复。因为该集合体系有索引。
2. ArrayList:数组结构.线程不同步，效率高,操作方法相同，都是可变长度数组。ArrayList是50%延长

线程同步的方式：

List list = Collections.synchronizedList(new ArraydList(...));

1. Vector: 数组结构.线程同步，效率低,而Vector是100%延长。

Vector中有[枚举]：

for (Enumeration<E> e = v.elements(); e.hasMoreElements();)

System.out.println(e.nextElement());

1. LinkedList: 链表结构，对于数据的增删改查比较方便。

LinkedList特有方法：

addFirst()addLast()getFirst()getLast()removeFirst()removeLast()【1.6后的替代方法】

offer()offerFirst()peek()peekFirst()poll()pollFirst()

之前的方法返回NoSuchElementException,替代方法此方法返回NULL

List的方法：List结合特有的迭代器。ListIterator是Iterator的子接口

Iterator只能使用判断，获取，删除元素使用ListIterator可以有其他更多的功能

增：add(index,element) addAll(index,collection)

删remove(index) clear()

改set(index,element)

查get(index) subList(fromindex,endindex) listIterator()

indexOf(Object);

判断contains() containsAll(collection) isEmpty() equals()

1. **Set.** 元素是无序（存入和取出顺序不一定一致），元素不可以重复

1）HashSet. 底层数据结构是哈希值，通过hashCode和equals方法来判断元素是否重复

2）TreeSet.可以对set集合中的元素进行排序。底层数据结构是二叉树（红黑树）。

TreeSet第一种排序：让元素自身具备比较性，Comparable接口覆写compareTo方法return 0

TreeSet第二种排序：让集合自身具备比较性。定义一个比较器。实现Comparator借口，覆盖compare方法。

# Map. 该集合存储键值对。一对一对往里存。而且要保证键的唯一性, Set集合底层就是Map集合。当发现数据有映射关系是使用Map集合

1. **HashMap.** 底层是哈希表数据结构,允许使用null键和null值,该集合是不同步的,效率高。
2. **TreeMap.** 底层是二叉树结构，线程不同步。可以用于给map集合中的键排序
3. **Hashtable.**底层是哈希表数据结构，线程同步的，不可以存在null键null值。效率低

Hashtable<String,Integer> ht=new Hashtable<String,Integer>();

String是key的类型，Integer是value的类型。

Map集合：该集合存储键值对。一对对往里存。而且要保证键的唯一性。

1.添加。put(K key,V value)//put方法返回原来在此键上定义的值

putAll(Map<? extends K,? extends V> m)

2.删除。clear()

3.判断。containsValue(Object value)containsKey(Object key) isEmpty()

Map集合的两种取出方式：

1.Set<K> keyset()：将Map集合中的键存入Set中,用Set中的迭代器。

2.Set<Map.Entry<k,v>> entrySet()：将map集合中的映射关系存入set集合中，而这个关系的数据类型就是Map.Entry

方法：equals()getKey()getValue()hashCode()setValue()

# Arrays

Arrays.asList(T) 将T转换成List集合

|  |  |  |
| --- | --- | --- |
| |  | | --- | | static<T> void | | [**sort**](mk:@MSITStore:E:\JAVA\JAVA_API文档1.6\JDK1.6%20API帮助文档.CHM::/java/util/Arrays.html#sort(T[], int, int, java.util.Comparator))(T[] a, int fromIndex, int toIndex, [Comparator](mk:@MSITStore:E:\JAVA\JAVA_API文档1.6\JDK1.6%20API帮助文档.CHM::/java/util/Comparator.html)<? super T> c) |

对指定数组升序排列

# Collections

\* 集合框架的工具类

\*

\* Collections中的集合工具类中的方法》

\* fill(list，Object) //将指定集合中的元素全部替换

\* reverse(list) //反转指定集合中的元素

\* replaceAll(List<T> list,T oldVal,T newVal) //将集合中所有oldVal替换成newVal

\* binarySearch() //传入可比较的集合和值来获取指定对象

\* sort(list,Comparator) //用户可以自己传入比较器

\* max(list,value,Comparator)

\* reverseOrder() //逆向返回一个迭代器