[**Map、Set、List、Queue、Stack的特点与用法**](http://www.cnblogs.com/yw-ah/p/5841369.html)

Collection       接口的接口   对象的集合   
├ List     **子接口**     按进入先后有序保存   可重复   
│├ LinkedList  接口实现类   链表   插入删除   没有同步   线程不安全   
│├ ArrayList   接口实现类   数组   随机访问   没有同步   线程不安全   
│└ Vector    接口实现类   数组    同步        线程安全   
│ 　 └ Stack  
└ Set   **子接口**      仅接收一次，并做内部排序

  ├ HashSet

  │ 　 └ LinkedHashSet  
  └ TreeSet

对于 List ，关心的是顺序， 它保证维护元素特定的顺序（允许有相同元素），使用此接口能够精确的控制每个元素插入的位置。用户能够使用索引（元素在 List 中的位置，类似于数组下标）来访问 List 中的元素。

对于 Set ，只关心某元素是否属于 Set （不 允许有相同元素 ），而不关心它的顺序。

Map                接口 键值对的集合   
├ Hashtable   接口实现类  同步    线程安全   
├ HashMap   接口实现类   没有同步    线程不安全

│├ LinkedHashMap

│└ WeakHashMap

├ TreeMap  
└ IdentifyHashMap

对于 Map ，最大的特点是键值映射，且为一一映射，键不能重复，值可以，所以是用键来索引值。 方法 put(Object key, Object value) 添加一个“值” ( 想要得东西 ) 和与“值”相关联的“键” (key) ( 使用它来查找 ) 。方法 get(Object key) 返回与给定“键”相关联的“值”。

Map 同样对每个元素保存一份，但这是基于 " 键 " 的， Map 也有内置的排序，因而不关心元素添加的顺序。如果添加元素的顺序对你很重要，应该使用 LinkedHashSet 或者 LinkedHashMap.

对于效率， Map 由于采用了哈希散列，查找元素时明显比 ArrayList 快。

更为精炼的总结：

Collection 是对象集合， Collection 有两个子接口 List 和 Set

List 可以通过下标 (1,2..) 来取得值，值可以重复

而 Set 只能通过游标来取值，并且值是不能重复的

ArrayList ， Vector ， LinkedList 是 List 的实现类

ArrayList 是线程不安全的， Vector 是线程安全的，这两个类底层都是由数组实现的

LinkedList 是线程不安全的，底层是由链表实现的

Map 是键值对集合

HashTable 和 HashMap 是 Map 的实现类     
HashTable 是线程安全的，不能存储 null 值     
HashMap 不是线程安全的，可以存储 null 值

Stack类：继承自Vector，实现一个后进先出的栈。提供了几个基本方法，push、pop、peak、empty、search等。

Queue接口：提供了几个基本方法，offer、poll、peek等。已知实现类有LinkedList、PriorityQueue等。