<http://blog.csdn.net/speedme/article/details/22398395>

版权声明：本文为博主原创文章，未经博主允许不得转载。

目录[(?)[+]](http://blog.csdn.net/speedme/article/details/22398395)

# Set,List,Map的区别

java集合的主要分为三种类型：

* Set（集）
* List（列表）
* Map（映射）

要深入理解集合首先要了解下我们熟悉的数组：

数组是大小固定的，并且同一个数组只能存放类型一样的数据（基本类型/引用类型），而JAVA集合可以存储和操作数目不固定的一组数据。 所有的JAVA集合都位于 java.util包中！ JAVA集合只能存放引用类型的的数据，不能存放基本数据类型。

简单说下集合和数组的区别：(参考文章：[《Thinking In Algorithm》03.数据结构之数组](http://blog.csdn.net/speedme/article/details/18180817" \t "_blank))

**[html]** [view plain](http://blog.csdn.net/speedme/article/details/22398395) [copy](http://blog.csdn.net/speedme/article/details/22398395)

[print?](http://blog.csdn.net/speedme/article/details/22398395)

1. **<span** style="font-family:Microsoft YaHei;font-size:12px;"**>**世间上本来没有集合,(只有数组参考C语言)但有人想要,所以有了集合
2. 有人想有可以自动扩展的数组,所以有了List
3. 有的人想有没有重复的数组,所以有了set
4. 有人想有自动排序的组数,所以有了TreeSet,TreeList,Tree\*\*
6. 而几乎有有的集合都是基于数组来实现的.
7. 因为集合是对数组做的封装,所以,数组永远比任何一个集合要快
9. 但任何一个集合,比数组提供的功能要多
11. 一：数组声明了它容纳的元素的类型，而集合不声明。这是由于集合以object形式来存储它们的元素。
13. 二：一个数组实例具有固定的大小，不能伸缩。集合则可根据需要动态改变大小。
15. 三：数组是一种可读/可写数据结构－－－没有办法创建一个只读数组。然而可以使用集合提供的ReadOnly方法，以只读方式来使用集合。该方法将返回一个集合的只读版本。**</span>**

Java所有“存储及随机访问一连串对象”的做法，array是最有效率的一种。

1、  
效率高，但容量固定且无法动态改变。  
array还有一个缺点是，无法判断其中实际存有多少元素，length只是告诉我们array的容量。

2、**[Java](http://lib.csdn.net/base/java" \o "Java 知识库" \t "_blank)**中有一个**Arrays类，专门用来操作array**。  
     arrays中拥有一组static函数，  
equals()：比较两个array是否相等。array拥有相同元素个数，且所有对应元素两两相等。  
fill()：将值填入array中。  
sort()：用来对array进行排序。  
binarySearch()：在排好序的array中寻找元素。  
System.arraycopy()：array的复制。

若撰写程序时不知道究竟需要多少对象，需要在空间不足时自动扩增容量，则需要使用容器类库，array不适用。所以就要用到集合。

那我们开始讨论java中的集合。

集合分类：

Collection：List、Set  
Map：HashMap、HashTable

## 1.1 Collection接口

Collection是最基本的集合接口，声明了适用于JAVA集合（只包括Set和List）的通用方法。 Set 和List 都继承了Conllection,Map。

### 1.1.1  Collection接口的方法：

**[html]** [view plain](http://blog.csdn.net/speedme/article/details/22398395) [copy](http://blog.csdn.net/speedme/article/details/22398395)

[print?](http://blog.csdn.net/speedme/article/details/22398395)

1. **<span** style="font-weight: normal;"**>**boolean add(Object o)      ：向集合中加入一个对象的引用
3. void clear()：删除集合中所有的对象，即不再持有这些对象的引用
5. boolean isEmpty()    ：判断集合是否为空
7. boolean contains(Object o) ： 判断集合中是否持有特定对象的引用
9. Iterartor iterator()  ：返回一个Iterator对象，可以用来遍历集合中的元素
11. boolean remove(Object o) ：从集合中删除一个对象的引用
13. int size()       ：返回集合中元素的数目
15. Object[] toArray()    ： 返回一个数组，该数组中包括集合中的所有元素 **</span>**

关于：Iterator() 和toArray() 方法都用于集合的所有的元素，前者返回一个Iterator对象，后者返回一个包含集合中所有元素的数组。

### 1.1.2  Iterator接口声明了如下方法：

**[html]** [view plain](http://blog.csdn.net/speedme/article/details/22398395) [copy](http://blog.csdn.net/speedme/article/details/22398395)

[print?](http://blog.csdn.net/speedme/article/details/22398395)

1. hasNext()：判断集合中元素是否遍历完毕，如果没有，就返回true
3. next() ：返回下一个元素
5. remove()：从集合中删除上一个有next()方法返回的元素。

## 1.2  Set(集合)

Set是最简单的一种集合。集合中的对象不按特定的方式排序，并且没有重复对象。 Set接口主要实现了两个实现类：

* HashSet： HashSet类按照哈希算法来存取集合中的对象，存取速度比较快
* TreeSet ：TreeSet类实现了SortedSet接口，能够对集合中的对象进行排序。

Set 的用法：存放的是对象的引用，没有重复对象

1. Set set=**new** HashSet();
3. String s1=**new** String("hello");
5. String s2=s1;
7. String s3=**new** String("world");
9. set.add(s1);
11. set.add(s2);
13. set.add(s3);
15. System.out.println(set.size());//打印集合中对象的数目 为 2。

Set 的 add()方法是如何判断对象是否已经存放在集合中？

1. **boolean** isExists=**false**;
3. Iterator iterator=set.iterator();
5. **while**(it.hasNext())           {
7. String oldStr=it.next();
9. **if**(newStr.equals(oldStr)){
11. isExists=**true**;
13. }
15. }

Set的功能方法

Set具有与Collection完全一样的接口，因此没有任何额外的功能，不像前面有两个不同的List。实际上Set就是Collection,只 是行为不同。(这是继承与多态思想的典型应用：表现不同的行为。)Set不保存重复的元素(至于如何判断元素相同则较为负责)

Set : 存入Set的每个元素都必须是唯一的，因为Set不保存重复元素。加入Set的元素必须定义equals()方法以确保对象的唯一性。Set与Collection有完全一样的接口。Set接口不保证维护元素的次序。

## HashSet：为快速查找设计的Set。存入HashSet的对象必须定义hashCode()。

## TreeSet： 保存次序的Set, 底层为树结构。使用它可以从Set中提取有序的序列。

## LinkedHashSet：具有HashSet的查询速度，且内部使用链表维护元素的顺序(插入的次序)。于是在使用迭代器遍历Set时，结果会按元素插入的次序显示。

## 1.3  List(列表)

List的特征是其元素以线性方式存储，集合中可以存放重复对象。

List接口主要实现类包括：（参考文章：[ArrayList与LinkedList的区别](http://blog.csdn.net/speedme/article/details/8199418" \t "_blank)）

* ArrayList() : 代表长度可以改变得数组。可以对元素进行随机的访问，向ArrayList()中插入与删除元素的速度慢。
* LinkedList(): 在实现中采用链表数据结构。插入和删除速度快，访问速度慢。

对于List的随机访问来说，就是只随机来检索位于特定位置的元素。 List 的 get(int index) 方法放回集合中由参数index指定的索引位置的对象，下标从“0” 开始。最基本的两种检索集合中的所有对象的方法：   
  
  
      1： for循环和get()方法：

1. **for**(**int** i=0; i<list.size();i++){
3. System.out.println(list.get(i));
5. }

2： 使用 迭代器（Iterator）:

1. Iterator it=list.iterator();
3. **while**(it.hashNext()){
5. System.out.println(it.next());
7. }

List的功能方法 

实际上有两种List：一种是基本的ArrayList,其优点在于随机访问元素，另一种是更强大的LinkedList,它并不是为快速随机访问设计的，而是具有一套更通用的方法。

## List：次序是List最重要的特点：它保证维护元素特定的顺序。List为Collection添加了许多方法，使得能够向List中间插入与移除元素(这只推 荐LinkedList使用。)一个List可以生成ListIterator,使用它可以从两个方向遍历List,也可以从List中间插入和移除元 素。

## ArrayList：由数组实现的List。允许对元素进行快速随机访问，但是向List中间插入与移除元素的速度很慢。ListIterator只应该用来由后向前遍历 ArrayList,而不是用来插入和移除元素。因为那比LinkedList开销要大很多。

## LinkedList ：对顺序访问进行了优化，向List中间插入与删除的开销并不大。随机访问则相对较慢。(使用ArrayList代替。)还具有下列方 法：addFirst(), addLast(), getFirst(), getLast(), removeFirst() 和 removeLast(), 这些方法 (没有在任何接口或基类中定义过)使得LinkedList可以当作堆栈、队列和双向队列使用。

## 1.4 Map(映射)

Map 是一种把键对象和值对象映射的集合，它的每一个元素都包含一对键对象和值对象。 Map没有继承于Collection接口 从Map集合中检索元素时，只要给出键对象，就会返回对应的值对象。

Map 的常用方法：

1 添加，删除操作：

**[html]** [view plain](http://blog.csdn.net/speedme/article/details/22398395) [copy](http://blog.csdn.net/speedme/article/details/22398395)

[print?](http://blog.csdn.net/speedme/article/details/22398395)

1. Object put(Object key, Object value)： 向集合中加入元素
2. Object remove(Object key)： 删除与KEY相关的元素
3. void putAll(Map t)：  将来自特定映像的所有元素添加给该映像
4. void clear()：从映像中删除所有映射

2 查询操作：

Object get(Object key)：获得与关键字key相关的值 。Map集合中的键对象不允许重复，也就说，任意两个键对象通过equals()方法比较的结果都是false.，但是可以将任意多个键独享映射到同一个值对象上。

Map的功能方法

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”。可以用containsKey()和containsValue()[**测试**](http://lib.csdn.net/base/softwaretest)Map中是否包含某个“键”或“值”。 标准的Java类库中包含了几种不同的Map：HashMap, TreeMap, LinkedHashMap, WeakHashMap, IdentityHashMap。它们都有同样的基本接口Map，但是行为、效率、排序策略、保存对象的生命周期和判定“键”等价的策略等各不相同。

执行效率是Map的一个大问题。看看get()要做哪些事，就会明白为什么在ArrayList中搜索“键”是相当慢的。而这正是HashMap提高速 度的地方。HashMap使用了特殊的值，称为“散列码”(hash code)，来取代对键的缓慢搜索。“散列码”是“相对唯一”用以代表对象的int值，它是通过将该对象的某些信息进行转换而生成的。所有Java对象都 能产生散列码，因为hashCode()是定义在基类Object中的方法。

HashMap就是使用对象的hashCode()进行快速查询的。此方法能够显着提高性能。

Map : 维护“键值对”的关联性，使你可以通过“键”查找“值”

HashMap：Map基于散列表的实现。插入和查询“键值对”的开销是固定的。可以通过构造器设置容量capacity和负载因子load factor，以调整容器的性能。

LinkedHashMap： 类似于HashMap，但是迭代遍历它时，取得“键值对”的顺序是其插入次序，或者是最近最少使用(LRU)的次序。只比HashMap慢一点。而在迭代访问时发而更快，因为它使用链表维护内部次序。

TreeMap ： 基于红黑树[**数据结构**](http://lib.csdn.net/base/datastructure)的实现。查看“键”或“键值对”时，它们会被排序(次序由Comparabel或Comparator决定)。TreeMap的特点在 于，你得到的结果是经过排序的。TreeMap是唯一的带有subMap()方法的Map，它可以返回一个子树。

WeakHashMao ：弱键(weak key)Map，Map中使用的对象也被允许释放: 这是为解决特殊问题设计的。如果没有map之外的引用指向某个“键”，则此“键”可以被垃圾收集器回收。

IdentifyHashMap： : 使用==代替equals()对“键”作比较的hash map。专为解决特殊问题而设计。

## 1.4 区别

### ****1.4.1、Collection 和 Map 的区别****

容器内每个为之所存储的元素个数不同。  
Collection类型者，每个位置只有一个元素。  
Map类型者，持有 key-value pair，像个小型**[数据库](http://lib.csdn.net/base/mysql" \o "MySQL知识库" \t "_blank)**。

### ****1.4.2、各自旗下的子类关系****

**Collection**     --List：将以特定次序存储元素。所以取出来的顺序可能和放入顺序不同。  
           --ArrayList / LinkedList / Vector  
     --Set ： 不能含有重复的元素  
           --HashSet / TreeSet  
**Map**  
     --HashMap  
     --HashTable  
     --TreeMap

### ****1.4.3、其他特征****

List，Set，Map将持有对象一律视为Object型别。

Collection、List、Set、Map都是接口，不能实例化。  
    继承自它们的 ArrayList, Vector, HashTable, HashMap是具象class，这些才可被实例化。  
vector容器确切知道它所持有的对象隶属什么型别。vector不进行边界检查。

总结

1. 如果涉及到堆栈，队列等操作，应该考虑用List，对于需要快速插入，删除元素，应该使用LinkedList，如果需要快速随机访问元素，应该使用ArrayList。  
2. 如果程序在单线程环境中，或者访问仅仅在一个线程中进行，考虑非同步的类，其效率较高，如果多个线程可能同时操作一个类，应该使用同步的类。  
3. 在除需要排序时使用TreeSet,TreeMap外,都应使用HashSet,HashMap,因为他们 的效率更高。  
4. 要特别注意对哈希表的操作，作为key的对象要正确复写equals和hashCode方法。  
   
5. 容器类仅能持有对象引用（指向对象的指针），而不是将对象信息copy一份至数列某位置。一旦将对象置入容器内，便损失了该对象的型别信息。  
6. 尽量返回接口而非实际的类型，如返回List而非ArrayList，这样如果以后需要将ArrayList换成LinkedList时，客户端代码不用改变。这就是针对抽象编程。

注意：

1、Collection没有get()方法来取得某个元素。只能通过iterator()遍历元素。

2、Set和Collection拥有一模一样的接口。

3、List，可以通过get()方法来一次取出一个元素。使用数字来选择一堆对象中的一个，get(0)...。(add/get)

4、一般使用ArrayList。用LinkedList构造堆栈stack、队列queue。

5、Map用 put(k,v) / get(k)，还可以使用containsKey()/containsValue()来检查其中是否含有某个key/value。

      HashMap会利用对象的hashCode来快速找到key。

6、Map中元素，可以将key序列、value序列单独抽取出来。

使用keySet()抽取key序列，将map中的所有keys生成一个Set。

使用values()抽取value序列，将map中的所有values生成一个Collection。

为什么一个生成Set，一个生成Collection？那是因为，key总是独一无二的，value允许重复。

Reference:

http://developer.51cto.com/art/201309/410205\_all.htm

http://billy-lee.iteye.com/blog/356398