

<https://blog.csdn.net/FengGLA/article/details/54869858?utm_source=blogxgwz0>

<https://www.cnblogs.com/danbing/p/5034108.html>

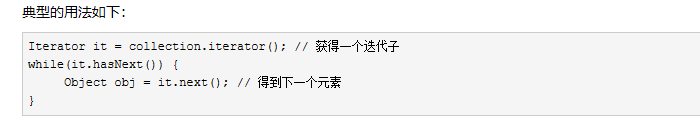
当new一个对象出来的时候，这个对象会产生一个this的引用，这个this引用指向对象自身。如果new出来的对象是一个子类对象的话，那么这个子类对象里面还会有一个super引用，这个super指向当前对象里面的父对象。所以相当于程序里面有一个this，this指向对象自己，还有一个super，super指向当前对象里面的父对象。

**Collection接口**

Collection是最基本的集合接口，一个Collection代表一组Object，即Collection的元素（Elements）。

所有实现Collection接口的类都必须提供两个标准的构造函数：无参数的构造函数用于创建一个空的Collection，有一个 Collection参数的构造函数用于创建一个新的Collection，这个新的Collection与传入的Collection有相同的元素。后一个构造函数允许用户复制一个Collection。

如何遍历Collection中的每一个元素？不论Collection的实际类型如何，它都支持一个iterator()的方法，该方法返回一个迭代器，使用该迭代器即可遍历访问Collection中每一个元素，***通过Iterator的遍历是无序的***。



List接口

List是有序的Collection，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，这类似于Java的数组。

除了具有Collection接口必备的iterator()方法外，List还提供一个listIterator()方法，返回一个 ListIterator接口，和标准的Iterator接口相比，ListIterator多了一些add()之类的方法，允许添加，删除，设定元素， 还能向前或向后遍历。

实现List接口的常用类有LinkedList，ArrayList，Vector和Stack。

LinkedList类

LinkedList实现了List接口，允许null元素。此外LinkedList提供额外的get，remove，insert方法。这些操作使LinkedList可被用作堆栈（stack），队列（queue）或双向队列（deque）。

注意LinkedList没有同步性。如果多个线程同时访问一个LinkedList，则必须自己实现访问同步。另一种解决方法是在创建List时构造一个同步的List：List list = Collections.synchronizedList(new LinkedList(...));

ArrayList类

ArrayList实现了大小可变的数组。它允许所有元素，包括null。ArrayList没有同步性。

size，isEmpty，get，set方法运行时间为常数。但是add方法开销为分摊的常数，添加n个元素需要O(n)的时间。其他的方法运行时间为线性。

每个ArrayList实例都有一个容量（Capacity），即用于存储元素的数组的大小。这个容量可随着不断添加新元素而自动增加，但是增长算法并没有定义。当需要插入大量元素时，在插入前可以调用ensureCapacity方法来增加ArrayList的容量以提高插入效率。

Vector类

Vector非常类似ArrayList，但是Vector是同步的。由Vector创建的Iterator，虽然和 ArrayList创建的Iterator是同一接口，但是，因为Vector是同步的，当一个Iterator被创建而且正在被使用，另一个线程改变了 Vector的状态（例如，添加或删除了一些元素），这时调用Iterator的方法时将抛出 ConcurrentModificationException，因此必须捕获该异常。

Stack 类

Stack继承自Vector，实现一个后进先出的堆栈。Stack提供5个额外的方法使得Vector得以被当作堆栈使用。基本的push和pop 方法，还有peek方法得到栈顶的元素，empty方法测试堆栈是否为空，search方法检测一个元素在堆栈中的位置。Stack刚创建后是空栈。

Vector、ArrayList和LinkedList比较

1.Vector是线程同步的，所以它也是线程安全的，而ArrayList和LinkedList是非线程安全的。如果不考虑到线程的安全因素，一般用ArrayList和LinkedList效率比较高。

2.ArrayList和Vector是实现了基于动态数组的数据结构，LinkedList基于链表的数据结构。

3.如果集合中的元素的数目大于目前集合数组的长度时，Vector增长率为目前数组长度的100%,而ArrayList增长率为目前数组长度的50%.如果在集合中使用数据量比较大的数据，用vector有一定的优势；反之，用ArrayList有优势。

3.如果查找一个指定位置的数据，Vector和ArrayList使用的时间是相同的，花费时间为O(1)，而LinkedList需要遍历查找，花费时间为O(i)，效率不如前两者。

4.而如果移动、删除一个指定位置的数据花费的时间为0(n-i)n为总长度，这个时候就应该考虑使用LinkedList,因为它移动一个指定位置的数据所花费的时间为0(1)。

5.对于在指定位置插入数据，LinedList比较占优势，因为ArrayList要移动数据。

Set接口

Set是一种不包含重复的元素的Collection，即任意的两个元素e1和e2都有e1.equals(e2)=false，Set最多有一个null元素。

很明显，Set的构造函数有一个约束条件，传入的Collection参数不能包含重复的元素。

请注意：必须小心操作可变对象（Mutable Object）。如果一个Set中的可变元素改变了自身状态导致Object.equals(Object)=true将导致一些问题。

Map接口

请注意，Map没有继承Collection接口，Map提供key到value的映射。一个Map中不能包含相同的key，每个key只能映射一个 value。

Map接口提供3种集合的视图，Map的内容可以被当作一组key集合，一组value集合，或者一组key-value映射。

Hashtable类

Hashtable继承Map接口，实现一个key-value映射的哈希表。任何非空（non-null）的对象都可作为key或者value。

添加数据使用put(key, value)，取出数据使用get(key)，这两个基本操作的时间开销为常数。

Hashtable通过initial capacity和load factor两个参数调整性能。通常缺省的load factor 0.75较好地实现了时间和空间的均衡。增大load factor可以节省空间但相应的查找时间将增大，这会影响像get和put这样的操作。

由于作为key的对象将通过计算其散列函数来确定与之对应的value的位置，因此任何作为key的对象都必须实现hashCode和equals方法。hashCode和equals方法继承自根类Object。

Hashtable是同步的。

HashMap类

HashMap和Hashtable类似，不同之处在于HashMap是非同步的，并且允许null，即null value和null key。但是将HashMap视为Collection时（values()方法可返回Collection），其迭代子操作时间开销和HashMap 的容量成比例。因此，如果迭代操作的性能相当重要的话，不要将HashMap的初始化容量设得过高，或者load factor过低。

TreeMap类

HashMap通过hashcode对其内容进行快速查找，无序的；而TreeMap中所有的元素都保持着某种固定的顺序，有序的。

在Map 中插入、删除和定位元素，HashMap 是最好的选择。但如果您要按自然顺序或自定义顺序遍历键，那么TreeMap会更好。使用HashMap要求添加的键类明确定义了hashCode()和 equals()的实现。

TreeMap没有调优选项，因为该树总处于平衡状态。

WeakHashMap类

WeakHashMap是一种改进的HashMap，它对key实行“弱引用”，如果一个key不再被外部所引用，那么该key可以被GC回收。

总结

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。

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。

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。

使用Map时，查找、更新、删除、新增最好使用HashMap或HashTable；对Map进行自然顺序或自定义键顺序遍历时，最好使用TreeMap;

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。

MySQL路径

C:\Program Files\MySQL\MySQL Server 8.0\bin

