集合中Iterable与Iterator迭代器介绍

# java.lang.Iterable<T>接口

**java.lang.Iterable<T>接口**中只有一个方法：

**Iterator<T>** iterator() 返回一个在一组 T 类型的元素上进行迭代的**迭代器**。

(注意：Iterable<T>接口存在java.lang包中，不是java.util包)

public interface Iterable<T>：实现这个接口允许对象成为 **"foreach" 语句**的目标。

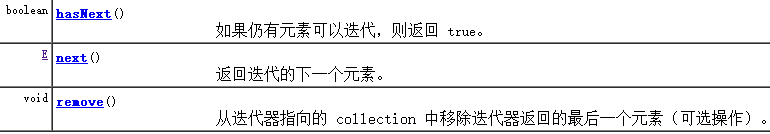
**java.util.Collection<E>接口**继承了java.lang.Iterable<T>接口，而**Collection接口**Collection 层次结构中的根接口；说明集合体系都具有迭代器，都可以使用**"foreach" 语句。**

# java.util.Iterator<E>接口：迭代器接口

java.util.**Iterator<E>:**对 collection 进行迭代的迭代器。迭代器取代了 Java Collections Framework 中的 **Enumeration**。迭代器与枚举有两点不同：

迭代器允许调用者利用定义良好的语义在迭代期间从迭代器所指向的 collection 移除元素；方法名称得到了改进。

java.util.**Iterator<E>中定义了三种方法：**



## hasNext()

boolean **hasNext**() 如果仍有元素可以迭代，则返回 true。

## next()

E **next**() 返回迭代的下一个元素。

抛出：NoSuchElementException - 没有元素可以迭代。

## remove()

void **remove**() 从迭代器指向的 collection 中移除迭代器返回的最后一个元素（可选操作）。**每次调用next 只能调用一次此方法。**如果进行迭代时用调用此方法之外的其他方式修改了该迭代器所指向的 collection，则迭代器的行为是不确定的。

抛出：

UnsupportedOperationException - 如果迭代器不支持 remove 操作。

IllegalStateException - 如果尚未调用 next 方法，或者在上一次调用 next 方法之后已经调用了 remove 方法。

# 子接口java.util.ListIterator<E>

public interface ListIterator<E> extends Iterator<E>

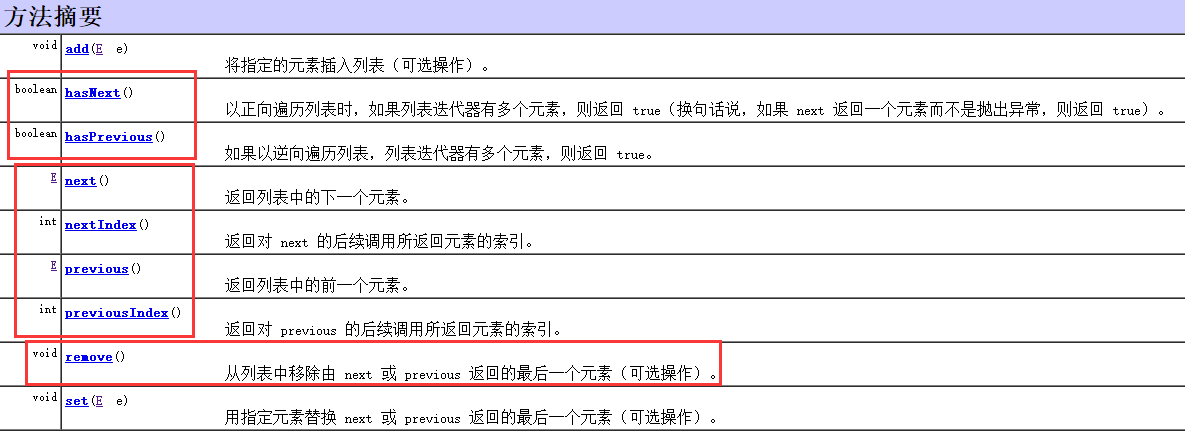
**列表迭代器**，允许程序员按**任一方向遍历列表**、*迭代期间修改列表*，并获得迭代器在列表中的当前位置。ListIterator 没有当前元素；它的**光标位置**始终位于**调用 previous() 所返回的元素和调用 next() 所返回的元素之间**。长度为 n 的列表的迭代器有 n+1 个可能的指针位置。

**当前光标位置示例：**



注意，**remove**() 和 **set**(Object) 方法不是 根据**光标位置**定义的；它们是根据对调用 next() 或 previous() 所返回的**最后一个元素**的操作定义的。

既然ListIterator<E>继承Iterator<E>，那么它必然继承Iterator<E>的三个方法。



# 实现类java.util.Scanner：扫描类

java.lang.Object

继承者 java.util.Scanner

java.util.Scanner该扫描类：

具有hasNext方法、next、remove方法。