一、使用Iterator迭代器遍历集合元素

1、terator对象称为迭代器(设计模式的一种)，主要用于遍历 Collection 集合中的元素

2、GOF给迭代器模式的定义为：提供一种方法访问一个容器(container)对象中各个元素，而又不需暴露该对象的内部细节。迭代器模式，就是为容器而生

3、Collection接口继承了java.lang.Iterable接口，该接口有一个iterator()方法，那么所有实现了Collection接口的集合类都有一个iterator()方法，用以返回一个实现了Iterator接口的对象。

4、Iterator仅用于遍历集合，Iterator 本身并不提供承装对象的能力。如果需要创建Iterator

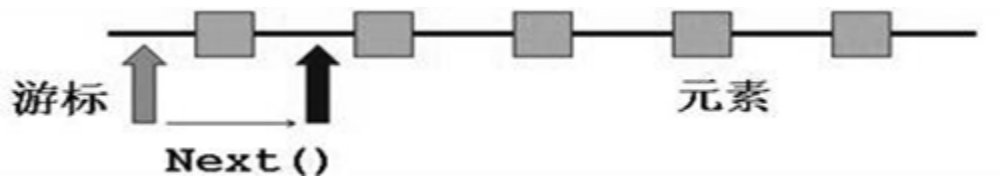
对象，则必须有一个被迭代的集合

5、集合对象每次调用iterator()方法都得到一个全新的迭代器对象，默认游标都在集合的第一个元素之前。

二、Iterator接口的方法

Method Summary

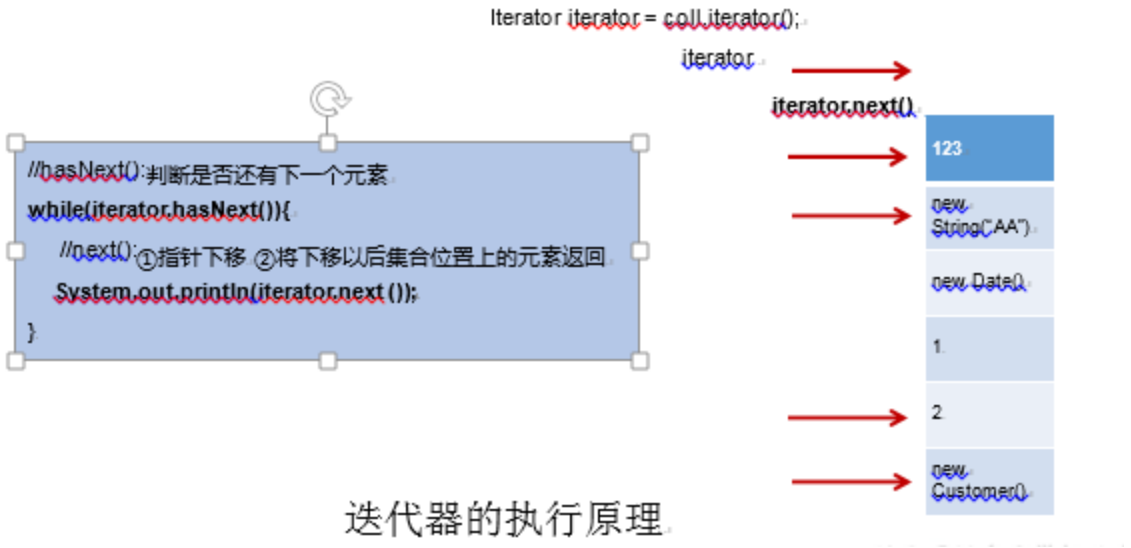
boolean	**hasNext**() Returns true if the iteration has more elements.
E	**next**() Returns the next element in the iteration.
void	**remove**() Removes from the underlying collection the last element returned by the iterator (optional operation).



三、集合元素的遍历操作

1、在调用next（）方法之前，一定要用hashNext方法进行检测，若不调用，且下一条记录无效，直接调用next（）方法会抛出NoSuchElementException异常

2、迭代器的执行原理



迭代器的执行原理

3.内部定义了remove(),可以在遍历的时候，删除集合中的元素。此方法不同于集合直接调用remove()

```
import org.junit.Test;

import java.util.ArrayList;
import java.util.Collection;
import java.util.Iterator;

/**
 * 集合元素的遍历操作，使用迭代器Iterator接口
 * 1.内部的方法：hasNext() 和 next()
 * 2.集合对象每次调用iterator()方法都得到一个全新的迭代器对象，
 * 默认游标都在集合的第一个元素之前。
 * 3.内部定义了remove(),可以在遍历的时候，删除集合中的元素。此方法不同于集合直接调用remove()
 *
 * @author shkstart
 * @create 2019 上午 10:44
```

```java
 */
public class IteratorTest {

 @Test
 public void test1(){
 Collection coll = new ArrayList();
 coll.add(123);
 coll.add(456);
 coll.add(new Person("Jerry",20));
 coll.add(new String("Tom"));
 coll.add(false);

 Iterator iterator = coll.iterator();
 //方式一：
// System.out.println(iterator.next());
// System.out.println(iterator.next());
// System.out.println(iterator.next());
// System.out.println(iterator.next());
// System.out.println(iterator.next());
// //报异常：NoSuchElementException
// System.out.println(iterator.next());

 //方式二：不推荐
// for(int i = 0;i < coll.size();i++){
// System.out.println(iterator.next());
// }

 //方式三：推荐
 ////hasNext():判断是否还有下一个元素
 while(iterator.hasNext()){
 //next():①指针下移 ②将下移以后集合位置上的元素返回
 System.out.println(iterator.next());
 }

 }
```

```
51
52  @Test
53  public void test2(){
54
55  Collection coll = new ArrayList();
56  coll.add(123);
57  coll.add(456);
58  coll.add(new Person("Jerry",20));
59  coll.add(new String("Tom"));
60  coll.add(false);
61
62  //错误方式一:
63 // Iterator iterator = coll.iterator();
64 // while((iterator.next()) != null){
65 // System.out.println(iterator.next());
66 // }
67
68  //错误方式二:
69  //集合对象每次调用iterator()方法都得到一个全新的迭代器对象，默认游标
 都在集合的第一个元素之前。
70  while (coll.iterator().hasNext()){
71  System.out.println(coll.iterator().next());
72  }
73
74
75  }
76
77  //测试Iterator中的remove()
78  //如果还未调用next()或在上一次调用 next 方法之后已经调用了 remove
 方法，
79  // 再调用remove都会报IllegalStateException。
80  @Test
81  public void test3(){
82  Collection coll = new ArrayList();
83  coll.add(123);
84  coll.add(456);
```

```
85   coll.add(new Person("Jerry",20));
86   coll.add(new String("Tom"));
87   coll.add(false);
88
89   //删除集合中"Tom"
90   Iterator iterator = coll.iterator();
91   while (iterator.hasNext()){
92  // iterator.remove();
93   Object obj = iterator.next();
94   if("Tom".equals(obj)){
95   iterator.remove();
96  // iterator.remove();
97   }
98
99   }
100   //遍历集合
101   iterator = coll.iterator();
102   while (iterator.hasNext()){
103   System.out.println(iterator.next());
104   }
105   }
106  }
```

四、使用foreach循环遍历集合元素

jdk 5.0 新增了foreach循环，用于遍历集合、数组

遍历集合的底层依然调用了Iterator

```
1   @Test
2   public void test1(){
3   Collection coll = new ArrayList();
4   coll.add(123);
5   coll.add(456);
6   coll.add(new Person("Jerry",20));
7   coll.add(new String("Tom"));
8   coll.add(false);
9
10   //for(集合元素的类型 局部变量 : 集合对象)
```

```
11  //内部仍然调用了迭代器。
12  for(Object obj : coll){
13  System.out.println(obj);
14  }
15  }
16
17  @Test
18  public void test2(){
19  int[] arr = new int[]{1,2,3,4,5,6};
20  //for(数组元素的类型 局部变量 : 数组对象)
21  for(int i : arr){
22  System.out.println(i);
23  }
24  }
```