java中List的用法和实例详解

2010-05-05 11:26

|  |
| --- |
| List的用法 List包括List接口以及List接口的所有实现类。因为List接口实现了Collection接口，所以List接口拥有Collection接口提供的所有常用方法，又因为List是列表类型，所以List接口还提供了一些适合于自身的常用方法，如表1所示。 http://www.004.la/data/uploadfile/201003/20100321165252908.bmp 表1 List接口定义的常用方法及功能 从表1可以看出，List接口提供的适合于自身的常用方法均与索引有关，这是因为List集合为列表类型，以线性方式存储对象，可以通过对象的索引操作对象。 List接口的常用实现类有ArrayList和LinkedList，在使用List集合时，通常情况下声明为List类型，实例化时根据实际情况的需要，实例化为ArrayList或LinkedList，例如： List<String> l = new ArrayList<String>();// 利用ArrayList类实例化List集合 List<String> l2 = new LinkedList<String>();// 利用LinkedList类实例化List集合 1．add(int index, Object obj)方法和set(int index, Object obj)方法的区别 在使用List集合时需要注意区分add(int index, Object obj)方法和set(int index, Object obj)方法，前者是向指定索引位置添加对象，而后者是修改指定索引位置的对象，例如执行下面的代码： src\com\mwq\TestCollection.java关键代码： public static void main(String[] args) { String a = "A", b = "B", c = "C", d = "D", e = "E"; List<String> list = new LinkedList<String>(); list.add(a); list.add(e); list.add(d); list.set(1, b);// 将索引位置为1的对象e修改为对象b list.add(2, c);// 将对象c添加到索引位置为2的位置 Iterator<String> it = list.iterator(); while (it.hasNext()) { System.out.println(it.next()); } } 在控制台将输出如下信息： A B C D 因为List集合可以通过索引位置访问对象，所以还可以通过for循环遍历List集合，例如遍历上面代码中的List集合的代码如下： src\com\mwq\TestCollection.java关键代码： for (int i = 0; i < list.size(); i++) { System.out.println(list.get(i));// 利用get(int index)方法获得指定索引位置的对象 } src\com\mwq\TestCollection.java完整代码如下： package com.mwq; import java.util.ArrayList; import java.util.LinkedList; import java.util.Iterator; import java.util.List; public class TestCollection { public static void main(String[] args) { System.out.println("开始："); String a = "A", b = "B", c = "C", d = "D", e = "E"; List<String> list = new LinkedList<String>(); list.add(a); list.add(e); list.add(d); list.set(1, b);// 将索引位置为1的对象e修改为对象b list.add(2, c);// 将对象c添加到索引位置为2的位置 Iterator<String> it = list.iterator(); while (it.hasNext()) { System.out.println(it.next()); } //                 for (int i = 0; i < list.size(); i++) { //                       System.out.println(list.get(i));// 利用get(int index)方法获得指定索引位置的对象 //          } System.out.println("结束！"); } } 2．indexOf(Object obj)方法和lastIndexOf(Object obj)方法的区别 在使用List集合时需要注意区分indexOf(Object obj)方法和lastIndexOf(Object obj)方法，前者是获得指定对象的最小的索引位置，而后者是获得指定对象的最大的索引位置，前提条件是指定的对象在List集合中具有重复的对象，否则如果在List集合中有且仅有一个指定的对象，则通过这两个方法获得的索引位置是相同的，例如执行下面的代码： src\com\mwq\TestCollection.java关键代码： public static void main(String[] args) { String a = "A", b = "B", c = "C", d = "D", repeat = "Repeat"; List<String> list = new ArrayList<String>(); list.add(a);          // 索引位置为 0 list.add(repeat);      // 索引位置为 1 list.add(b);          // 索引位置为 2 list.add(repeat);      // 索引位置为 3  list.add(c);          // 索引位置为 4 list.add(repeat);      // 索引位置为 5 list.add(d);          // 索引位置为 6 System.out.println(list.indexOf(repeat)); System.out.println(list.lastIndexOf(repeat)); System.out.println(list.indexOf(b)); System.out.println(list.lastIndexOf(b)); } src\com\mwq\TestCollection.java完整代码如下： package com.mwq; import java.util.ArrayList; import java.util.List; public class TestCollection { public static void main(String[] args) { System.out.println("开始："); String a = "A", b = "B", c = "C", d = "D", repeat = "Repeat"; List<String> list = new ArrayList<String>(); list.add(a); // 索引位置为 0 list.add(repeat); // 索引位置为 1 list.add(b); // 索引位置为 2 list.add(repeat); // 索引位置为 3 list.add(c); // 索引位置为 4 list.add(repeat); // 索引位置为 5 list.add(d); // 索引位置为 6 System.out.println(list.indexOf(repeat)); System.out.println(list.lastIndexOf(repeat)); System.out.println(list.indexOf(b)); System.out.println(list.lastIndexOf(b)); System.out.println("结束！"); } } 在控制台将输出如下信息： 1 5 2 2 3．subList(int fromIndex, int toIndex)方法 在使用subList(int fromIndex, int toIndex)方法截取现有List集合中的部分对象生成新的List集合时，需要注意的是，新生成的集合中包含起始索引位置代表的对象，但是不包含终止索引位置代表的对象，例如执行下面的代码： src\com\mwq\TestCollection.java关键代码： public static void main(String[] args) { String a = "A", b = "B", c = "C", d = "D", e = "E"; List<String> list = new ArrayList<String>(); list.add(a);          // 索引位置为 0 list.add(b);          // 索引位置为 1 list.add(c);          // 索引位置为 2 list.add(d);          // 索引位置为 3 list.add(e);          // 索引位置为 4 list = list.subList(1, 3);// 利用从索引位置 1 到 3 的对象重新生成一个List集合 for (int i = 0; i < list.size(); i++) { System.out.println(list.get(i)); } } src\com\mwq\TestCollection.java完整代码： package com.mwq; import java.util.ArrayList; import java.util.List; public class TestCollection { public static void main(String[] args) { System.out.println("开始："); String a = "A", b = "B", c = "C", d = "D", e = "E"; List<String> list = new ArrayList<String>(); list.add(a); // 索引位置为 0 list.add(b); // 索引位置为 1 list.add(c); // 索引位置为 2 list.add(d); // 索引位置为 3 list.add(e); // 索引位置为 4 list = list.subList(1, 3);// 利用从索引位置 1 到 3 的对象重新生成一个List集合 for (int i = 0; i < list.size(); i++) { System.out.println(list.get(i)); } System.out.println("结束！"); } } 在控制台将输出如下信息： B C |

list中可以添加任何对象，我可以给你举个例子：

class Person{

.....

}

上面定义了一个Person类，下面看好如何使用List

Person p1=new Person();

Person p2=new Person();

List list=new ArrayList();

list.add(p1);

list.add(p2);//这里是将对象加入到list中

for(int i=0;i<list.size();i++){//利用循环，将person对象全部一一取出

Person p=(Person)list.get(i);//注意，这里一定要强制类型转换，因为List中取出的对象都是Object类型的，希望对你有所帮助

}