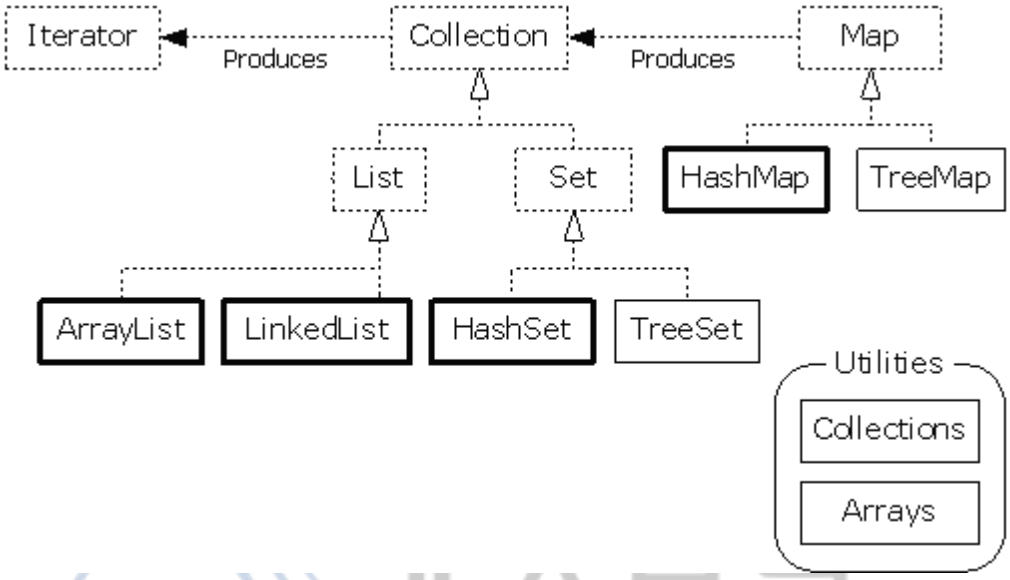
Collections 算法类

目 录

1. 概述

Java 集合框架由:对外提供的接口、接口的实现和对集合操作的算法三部分组成。



Java 集合框架将针对不同数据结构算法的实现都保存在工具类中，其中Arrays 类定义了用来操作数组的各种方法。Collections 类定义了一系列用于操作集合的工具方法，这些方法都
是静态的并且都是泛型方法，通过泛型保证了这些算法的类型安全。

Collections 类定义了很多工具方法，下面我根据这些方法的用途给大家讲解。

2. 对集合排序、交换的方法

Collections 类定义了常用的对集合排序、交换的方法，如表-1 所示。

表-1 Collections 类对集合排序、交换的方法

方法	说明
static void reverse(List list)	反转指定 List 集合中元素的顺序。
static void sort(List list)	根据元素的自然顺序 对指定List 集合按升序进行排序。
static void swap(List list, int i, int j)	在指定 List 集合的指定位置处交换元素。

- reverse()方法的作用是将指定 List 集合中的元素顺序反转。下面通过示例学习 reverse()方法的应用，代码如示例 1 所示。

示例 1：

```
import java.util.LinkedList;
import java.util.List;
import java.util.Collections;

public class CollectionsDemo {
    public static void main(String[] args) {
        List<String> list = new LinkedList<String>();
        list.add("one");
        list.add("two");
        list.add("three");
        for(String s:list){
            System.out.println(s);
        }
        System.out.println("执行 reverse 方法");
        Collections.reverse(list);
        for(String s:list){
           System.out.println(s);
        }
    }
}
```

示例 1 的运行效果如图 1 所示。

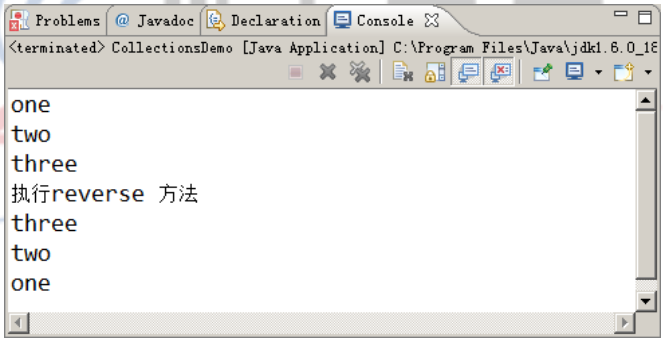


图 1 reverse()方法的使用

- sort 方法的作用是对指定 List 集合按升序进行排序。代码如示例 2 所示。

示例 2：

```
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;

public class CollectionsDemo {
    public static void main(String[] args) {
        List<Integer> list = new ArrayList<Integer>();
        list.add(1);
        list.add(3);
        list.add(2);
        for(Integer i:list){
            System.out.println(i);
        }
```

```
        System.out.println("执行 sort 方法");
        Collections.sort(list);
        for(Integer i:list){
            System.out.println(i);
        }
    }
}
```

示例 2 的运行效果如图 2 所示。

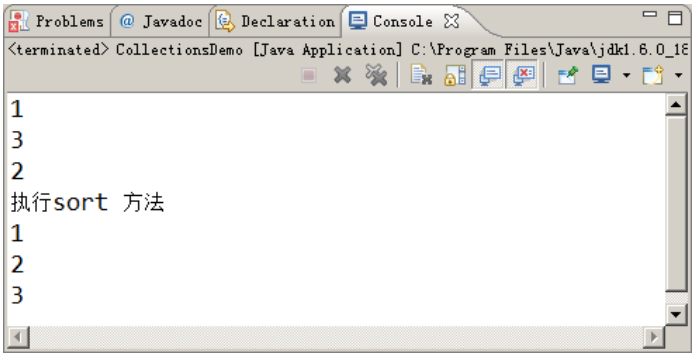


图 2 sort ()方法的使用

3. 对集合查找的方法

接下来我们将要学习 Collections 类中对集合查找、替换的方法，如表-2 所示。

表-2 Collections 类常用的查找方法

方法	说明
static int binarySearch(List list, T key)	使用二分查找算法查找指定List 集合，以获得指 定对象的索引位置。
static Object max(Collection coll)	根据元素的自然顺序，返回给定集合的最大 元素。
static Object min(Collection coll)	根据元素的自然顺序 返回给定集合的最小 元素。
static boolean replaceAll(List list, Object oldVal, Object newVal)	使用另一个值替换集合中出现的所有某一指 定值。

- binarySearch()方法表示使用二分查找算法查找出指定 List 集合中的指定元素，二分查找算法是针对以排序的集合，所以在调用 binarySearch()方法时，我们要使用 sort 对集合进行升序排序。如果没有对集合进行排序，则结果是不确定的。如果集合包含多个等于指定对象的元素，也是无法保证找到的是哪一个。

- max()方法是返回指定集合中由自然顺序决定的最大元素，这个指定的集合是不需要排序的。
- min()方法相反，它返回指定集合中由自然顺序决定的最小元素，同样这个集合也不要经过排序。

我们通过一个具体的示例来看一下这三个方法的用法，代码如示例 3 所示。

示例 3：

```
import java.util.Collections;
import java.util.LinkedList;
import java.util.List;

public class CollectionsDemo {
    public static void main(String[] args) {
        List<Integer> list = new LinkedList<Integer>();
        list.add(12);
        list.add(31);
        list.add(23);
        list.add(1);
        list.add(18);
        Integer max = Collections.max(list);
        Integer min = Collections.min(list);
        Collections.sort(list);
        int index = Collections.binarySearch(list, 12);
        System.out.println("12 的索引位置是："+index);
        System.out.println("集合中最大元素是："+max);
        System.out.println("集合中最小元素是："+min);
    }
}
```

示例 3 的运行效果如图 3 所示。

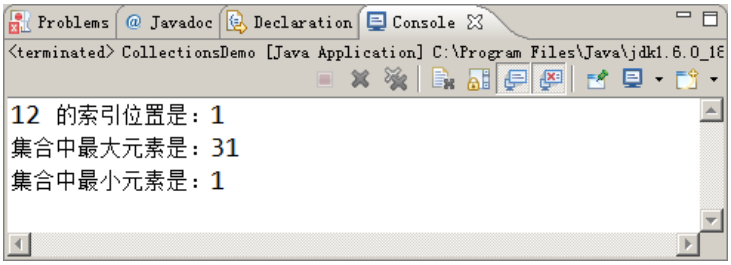


图 3 Collections 对集合查找方法的使用

其他 Collections 方法的使用，可以查阅 API。