每时每课 给你新机会

课工场 kgc.cn

大数据开发工程师系列

北京课工场教育科技有限公司 出品

◉ 视频课程 ◉ 案例素材 ◉ 交流社区 ◉ QQ讨论组

Java核心API编程

主 编 肖 睿 禹 晨 马 凌
副主编 邢远秀 谢海波 张 宇

中国水利水电出版社
www.waterpub.com.cn

大数据开发工程师系列

Java 核心 API 编程

主　编　肖　睿　禹　晨　马　凌
副主编　邢远秀　谢海波　张　宇

·北京·

内 容 提 要

本书深入探究 Java 高级实用技术的内容，从而进一步强化 Java 开发技能。主要内容包括集合框架、泛型、实用类、输入输出处理、多线程、Socket 网络编程、XML 解析等。

为保证最优学习效果，本书紧密结合实际应用，利用大量案例说明和实践，提炼含金量十足的开发经验。本书使用 Java 高级实用技术进行控制台程序开发，并配以完善的学习资源和支持服务，包括视频教程、案例素材下载、学习交流社区、讨论组等终身学习内容，为开发者带来全方位的学习体验，更多技术支持请访问课工场官网：www.kgc.cn。

图书在版编目（CIP）数据

Java核心API编程 / 肖睿，禹晨，马凌主编. -- 北京 : 中国水利水电出版社，2017.7
（大数据开发工程师系列）
ISBN 978-7-5170-5566-2

Ⅰ. ①J… Ⅱ. ①肖… ②禹… ③马… Ⅲ. ①JAVA语言－程序设计 Ⅳ. ①TP312.8

中国版本图书馆CIP数据核字(2017)第147203号

策划编辑：祝智敏　责任编辑：李　炎　加工编辑：于杰琼　封面设计：梁　燕

书　名	大数据开发工程师系列 Java核心API编程 Java HEXIN API BIANCHENG
作　者	主 编 肖 睿 禹 晨 马 凌 副主编 邢远秀 谢海波 张 宇
出版发行	中国水利水电出版社 （北京市海淀区玉渊潭南路 1 号 D 座 100038） 网 址：www.waterpub.com.cn E-mail：mchannel@263.net（万水） sales@waterpub.com.cn 电 话：（010）68367658（营销中心）、82562819（万水）
经　售	全国各地新华书店和相关出版物销售网点
排　版	北京万水电子信息有限公司
印　刷	北京泽宇印刷有限公司
规　格	184mm × 260mm　16 开本　12 印张　263 千字
版　次	2017 年 7 月第 1 版　2017 年 7 月第 1 次印刷
印　数	0001—3000 册
定　价	36.00 元

丛书编委会

前　言

丛书设计：

准备好了吗？进入大数据时代！大数据已经并将继续影响人类的方方面面。2015年8月31日，经李克强总理批准，国务院正式下发《关于印发促进大数据发展行动纲要的通知》，这是从国家层面正式宣告大数据时代的到来！企业资本则以BAT互联网公司为首，不断进行大数据创新，从而实现大数据的商业价值。本丛书根据企业人才实际需求，参考历史学习难度曲线，选取“Java + 大数据”技术集作为学习路径，旨在为读者提供一站式实战型大数据开发学习指导，帮助读者踏上由开发入门到大数据实战的互联网 + 大数据开发之旅！

丛书特点：

1．以企业需求为设计导向

满足企业对人才的技能需求是本丛书的核心设计原则，为此课工场大数据开发教研团队，通过对数百位BAT一线技术专家进行访谈、对上千家企业人力资源情况进行调研、对上万个企业招聘岗位进行需求分析，从而实现技术的准确定位，达到课程与企业需求的高契合度。

2．以任务驱动为讲解方式

丛书中的技能点和知识点都由任务驱动，读者在学习知识时不仅可以知其然，而且可以知其所以然，帮助读者融会贯通、举一反三。

3．以实战项目来提升技术

本丛书均设置项目实战环节，该环节综合运用书中的知识点，帮助读者提升项目开发能力。每个实战项目都设有相应的项目思路指导、重难点讲解、实现步骤总结和知识点梳理。

4．以互联网 + 实现终身学习

本丛书可通过使用课工场APP进行二维码扫描来观看配套视频的理论讲解和案例操作，同时课工场（www.kgc.cn）开辟教材配套版块，提供案例代码及案例素材下载。此外，课工场还为读者提供了体系化的学习路径、丰富的在线学习资源和活跃的学习社区，方便读者随时学习。

读者对象：

1．大中专院校的老师和学生

2．编程爱好者

3．初中级程序开发人员

4．相关培训机构的老师和学员

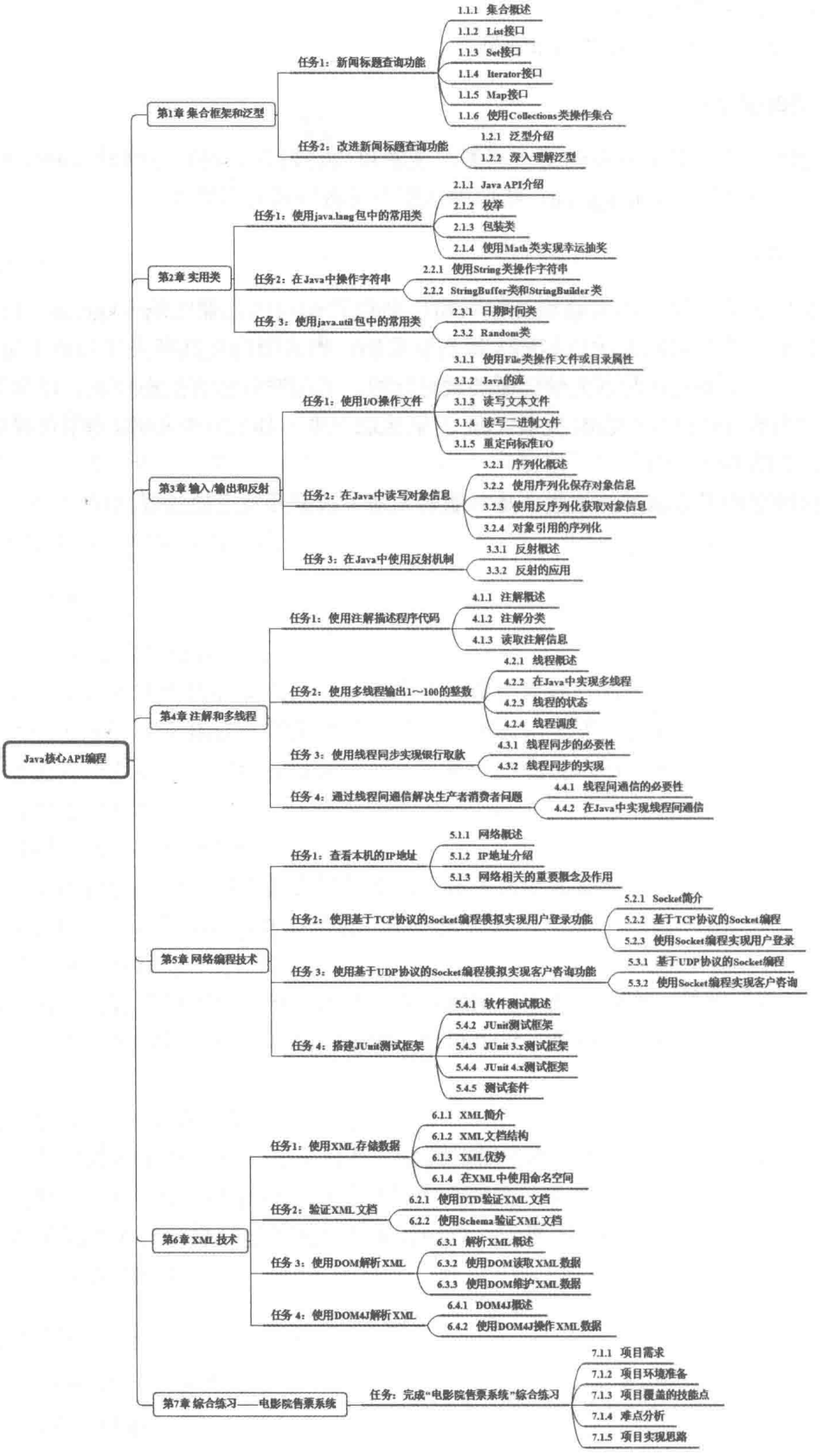
读者服务：

为解决本丛书中存在的疑难问题，读者可以访问课工场官方网站（www.kgc.cn），也可以发送邮件到 ke@kgc.cn，我们的客服专员将竭诚为您服务。

致谢：

本丛书是由课工场大数据开发教研团队研发编写的，课工场（kgc.cn）是北京大学旗下专注于互联网人才培养的高端教育品牌。作为国内互联网人才教育生态系统的构建者，课工场依托北京大学优质的教育资源，重构职业教育生态体系，以学员为本、以企业为基，构建教学大咖、技术大咖、行业大咖三咖一体的教学矩阵，为学员提供高端、靠谱、炫酷的学习内容！

感谢您购买本丛书，希望本丛书能成为您大数据开发之旅的好伙伴！


Java核心API编程
第1章 集合框架和泛型
任务1：新闻标题查询功能
1.1.1 集合概述
1.1.2 List接口
1.1.3 Set接口
1.1.4 Iterator接口
1.1.5 Map接口
1.1.6 使用Collections类操作集合
任务2：改进新闻标题查询功能
1.2.1 泛型介绍
1.2.2 深入理解泛型
第2章 实用类
任务1：使用java.lang包中的常用类
2.1.1 Java API介绍
2.1.2 枚举
2.1.3 包装类
2.1.4 使用Math类实现幸运抽奖
任务2：在Java中操作字符串
2.2.1 使用String类操作字符串
2.2.2 StringBuffer类和StringBuilder类
任务 3：使用java.util包中的常用类
2.3.1 日期时间类
2.3.2 Random类
第3章 输入/输出和反射
任务1：使用I/O操作文件
3.1.1 使用File类操作文件或目录属性
3.1.2 Java的流
3.1.3 读写文本文件
3.1.4 读写二进制文件
3.1.5 重定向标准I/O
任务2：在Java中读写对象信息
3.2.1 序列化概述
3.2.2 使用序列化保存对象信息
3.2.3 使用反序列化获取对象信息
3.2.4 对象引用的序列化
任务 3：在Java中使用反射机制
3.3.1 反射概述
3.3.2 反射的应用
第4章 注解和多线程
任务1：使用注解描述程序代码
4.1.1 注解概述
4.1.2 注解分类
4.1.3 读取注解信息
任务2：使用多线程输出1～100的整数
4.2.1 线程概述
4.2.2 在Java中实现多线程
4.2.3 线程的状态
4.2.4 线程调度
任务 3：使用线程同步实现银行取款
4.3.1 线程同步的必要性
4.3.2 线程同步的实现
任务 4：通过线程间通信解决生产者消费者问题
4.4.1 线程间通信的必要性
4.4.2 在Java中实现线程间通信
第5章 网络编程技术
任务1：查看本机的IP地址
5.1.1 网络概述
5.1.2 IP地址介绍
5.1.3 网络相关的重要概念及作用
任务2：使用基于TCP协议的Socket编程模拟实现用户登录功能
5.2.1 Socket简介
5.2.2 基于TCP协议的Socket编程
5.2.3 使用Socket编程实现用户登录
任务 3：使用基于UDP协议的Socket编程模拟实现客户咨询功能
5.3.1 基于UDP协议的Socket编程
5.3.2 使用Socket编程实现客户咨询
任务 4：搭建JUnit测试框架
5.4.1 软件测试概述
5.4.2 JUnit测试框架
5.4.3 JUnit 3.x测试框架
5.4.4 JUnit 4.x测试框架
5.4.5 测试套件
第6章 XML技术
任务1：使用XML存储数据
6.1.1 XML简介
6.1.2 XML文档结构
6.1.3 XML优势
6.1.4 在XML中使用命名空间
任务2：验证XML文档
6.2.1 使用DTD验证XML文档
6.2.2 使用Schema验证XML文档
任务 3：使用DOM解析XML
6.3.1 解析XML概述
6.3.2 使用DOM读取XML数据
6.3.3 使用DOM维护XML数据
任务 4：使用DOM4J解析XML
6.4.1 DOM4J概述
6.4.2 使用DOM4J操作XML数据
第7章 综合练习——电影院售票系统
任务：完成"电影院售票系统"综合练习
7.1.1 项目需求
7.1.2 项目环境准备
7.1.3 项目覆盖的技能点
7.1.4 难点分析
7.1.5 项目实现思路

目　录

第1章

集合框架和泛型

本章重点

※ List 接口及实现类
※ Map 接口及实现类
※ 泛型集合

本章目标

※ Iterator 接口
※ 泛型类、泛型接口

本章任务

学习本章，完成以下 2 个工作任务。记录学习过程中遇到的问题，可以通过自己的努力或访问 kgc.cn 解决。

任务 1：新闻标题查询功能

在新闻管理系统中，实现新闻标题信息的存储及查询功能。要求使用集合类存储新闻标题（包含 ID、名称、创建者）信息。

任务 2：改进新闻标题查询功能

改进任务 1，使用泛型集合存储新闻标题（包含 ID、名称、创建者）信息，输出新闻标题的总数量及每条新闻标题的名称。

任务 1　新闻标题查询功能

关键步骤如下：

- 创建集合对象，并添加数据。
- 统计新闻标题总数量。
- 输出新闻标题名称。

1.1.1　集合概述

开发应用程序时，如果想存储多个同类型的数据，可以使用数组来实现，但是使用数组存在如下一些明显缺陷：

- 数组长度固定不变，不能很好地适应元素数量动态变化的情况。
- 可通过数组名 .length 获取数组的长度，却无法直接获取数组中实际存储的元素个数。
- 数组采用在内存中分配连续空间的存储方式存储，根据元素信息查找时效率比较低，需要多次比较。

从以上分析可以看出数组在处理一些问题时存在明显的缺陷，针对数组的缺陷，Java 提供了比数组更灵活、更实用的集合框架，可大大提高软件的开发效率，并且不同的集合可适用于不同应用场合。

Java 集合框架提供了一套性能优良、使用方便的接口和类，它们都位于 java.util 包中，其主要内容及彼此之间的关系如图 1.1 所示。

从图 1.1 中可以看出，Java 的集合类主要由 Map 接口和 Collection 接口派生而来，其中 Collection 接口有两个常用的子接口，即 List 接口和 Set 接口，所以通常说 Java

集合框架由三大类接口构成（Map 接口、List 接口和 Set 接口）。本章讲解的主要内容就是围绕这三大类接口进行的。

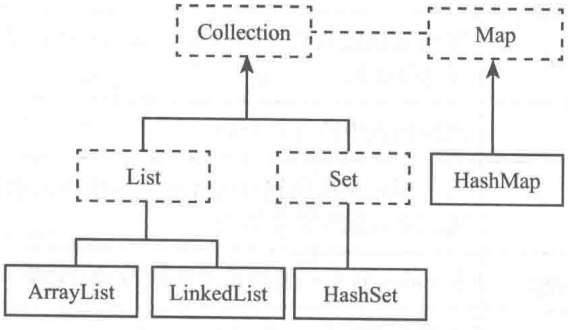


图 1.1　Java 集合框架图

注意：

虚线框表示接口或者抽象类，实线框表示开发中常用的实现类。

1.1.2　List 接口

Collection 接口是最基本的集合接口，可以存储一组不唯一、无序的对象。List 接口继承自 Collection 接口，是有序集合。用户可使用索引访问 List 接口中的元素，类似于数组。List 接口中允许存放重复元素，也就是说 List 可以存储一组不唯一、有序的对象。

List 接口常用的实现类有 ArrayList 和 LinkedList。

1. 使用 ArrayList 类动态存储数据

针对数组的一些缺陷，Java 集合框架提供了 ArrayList 集合类，对数组进行了封装，实现了长度可变的数组，而且和数组采用相同的存储方式，在内存中分配连续的空间，如图 1.2 所示，所以，经常称 ArrayList 为动态数组。但是它不等同于数组，ArrayList 集合中可以添加任何类型的数据，并且添加的数据都将转换成 Object 类型，而在数组中只能添加同一数据类型的数据。

0	1	2	3	4	5	
aaaa	dddd	cccc	aaaa	eeee	dddd	

图 1.2　ArrayList 存储方式示意图

ArrayList 类提供了很多方法用于操作数据，如表 1-1 中列出的是 ArrayList 类的常用方法。

表 1-1　ArrayList 类的常用方法

方　法	说　明
boolean add(Object o)	在列表的末尾添加元素 o，起始索引位置从 0 开始
void add(int index,Object o)	在指定的索引位置添加元素 o，索引位置必须介于 0 和列表中元素个数之间
int size()	返回列表中的元素个数
Object get(int index)	返回指定索引位置处的元素，取出的元素是 Object 类型，使用前需要进行强制类型转换
void set(int index,Object obj)	将 index 索引位置的元素替换为 obj 元素
boolean contains(Object o)	判断列表中是否存在指定元素 o
int indexOf(Object obj)	返回元素在集合中出现的索引位置
boolean remove(Object o)	从列表中删除元素 o
Object remove(int index)	从列表中删除指定位置的元素，起始索引位置从 0 开始

➲ 示例 1

使用 ArrayList 常用方法动态操作数据。

实现步骤：

（1）导入 ArrayList 类。

（2）创建 ArrayList 对象，并添加数据。

（3）判断集合中是否包含某元素。

（4）移除索引为 0 的元素。

（5）把索引为 1 的元素替换为其他元素。

（6）输出某个元素所在的索引位置。

（7）清空 ArrayList 集合中的数据。

（8）判断 ArrayList 集合中是否包含数据。

关键代码：

```
public static void main(String[] args){
  ArrayList list=new ArrayList();                    // ①
  list.add(" 张三丰 ");
  list.add(" 郭靖 ");
  list.add(" 杨过 ");
  // 判断集合中是否包含 " 李莫愁 "，对应步骤（3）
  System.out.println(list.contains(" 李莫愁 "));      // 输出 false
  // 把索引为 0 的数据移除，对应步骤（4）
  list.remove(0);                                    // ②
  System.out.println("----------------------");
  list.set(1," 黄蓉 ");                               // ③
```

```
    for (int i=0; i<list.size();i++){
        String name=(String)list.get(i);          //④
        System.out.println(name);
    }
    System.out.println("----------------------");
    System.out.println(list.indexOf(" 小龙女 "));          //⑤
    list.clear(); // 清空 list 中的数据
    System.out.println("----------------------");
    for(Object obj:list){
        String name=(String)obj;                  //⑥
        System.out.println(name);
    }
System.out.println(list.isEmpty());                    //⑦
}
```

输出结果如下所示：

```
false
----------------------
郭靖
黄蓉
----------------------
-1
----------------------
true
```

在示例 1 中，①的代码调用 ArrayList 的无参构造方法，创建集合对象。常用的 ArrayList 类的构造方法还有一个带参数的重载版本，即 ArrayList(int initialCapacity)，它构造一个具有指定初始容量的空列表。

② 的代码将 list 集合中索引为 0 的元素删除，list 集合的下标是从 0 开始，也就是删除了“张三丰”，集合中现有元素为“郭靖”和“杨过”。

③ 的代码将 list 集合中索引为 1 的元素替换为“黄蓉”，即将“杨过”替换为“黄蓉”，集合中现有元素为“郭靖”和“黄蓉”。

④ 的代码是使用 for 循环遍历集合，输出集合中的所有元素。list.get(i) 取出集合中索引为 i 的元素，并强制转换为 String 类型。

⑤ 的代码为输出元素“小龙女”所在的索引位置，因集合中没有该元素，所以输出结果为 -1。

⑥ 的代码是使用增强 for 循环遍历集合，输出集合中的所有元素。增强 for 循环的语法在 Java 基础课程中讲过，这里不再赘述。可以看出，遍历集合时使用增强 for 循环比普通的 for 循环在写法上更加简单方便，而且不用考虑下标越界的问题。

⑦ 的代码用来判断 list 集合是否为空，因为前面执行了 list.clear() 操作，所以集合已经为空，输出为 true。

注意：

① 调用 ArrayList 类的 add(Object obj) 方法时，添加到集合当中的数据将被转换为 Object 类型。

② 使用 ArrayList 类之前，需要导入相应的接口和类，代码如下：

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

示例 2

使用 ArrayList 集合存储新闻标题信息（包含 ID、名称、创建者），输出新闻标题的总数量及每条新闻标题的名称。

实现步骤：

（1）创建 ArrayList 对象，并添加数据。

（2）获取新闻标题的总数。

（3）遍历集合对象，输出新闻标题名称。

关键代码：

```
// 创建新闻标题对象，NewTitle 为新闻标题类
NewTitle car=new NewTitle(1, " 汽车 ", " 管理员 ");
NewTitle test=new NewTitle(2, " 高考 ", " 管理员 ");
// 创建存储新闻标题的集合对象
List newsTitleList=new ArrayList();
// 按照顺序依次添加新闻标题
newsTitleList.add(car);
newsTitleList.add(test);
// 获取新闻标题的总数
System.out.println(" 新闻标题数目为："+newsTitleList.size()+" 条 ");
// 遍历集合对象
System.out.println(" 新闻标题名称为：");
for(Object obj:newsTitleList){
   NewTitle title=(NewTitle)obj;
   System.out.println(title.getTitleName());
}
```

输出结果如图 1.3 所示。

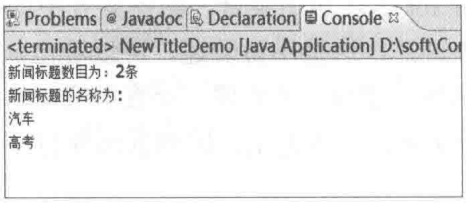


图 1.3　输出新闻标题信息

在示例 2 中，ArrayList 集合中存储的是新闻标题对象。在 ArrayList 集合中可以存储任何类型的对象。其中，代码 List newsTitleList=new ArrayList(); 是将接口 List 的引用指向了实现类 ArrayList 的对象，在编程中将接口的引用指向实现类的对象是 Java 实现多态的一种形式，也是软件开发中实现低耦合的方式之一，这样的用法可以大大提高程序的灵活性。随着编程经验的积累，对这个用法的理解会逐步加深。

ArrayList 因为可以使用索引来直接获取元素，所以其优点是遍历元素和随机访问元素的效率比较高。但是由于 ArrayList 采用了和数组相同的存储方式，在内存中分配连续的空间，因此在添加和删除非尾部元素时会导致后面所有元素的移动，这就造成在插入、删除等操作频繁的应用场景下用 ArrayList 会性能低下。所以数据操作频繁时，最好使用 LinkedList 存储数据。

2．使用 LinkedList 类动态存储数据

LinkedList 类是 List 接口的链接列表实现类。它支持实现所有 List 接口可选的列表的操作，并且允许元素值是任何数据，包括 null。

LinkedList 采用链表存储方式存储数据，如图 1.4 所示，优点在于插入、删除元素时效率比较高，但是 LinkedList 的查找效率很低。

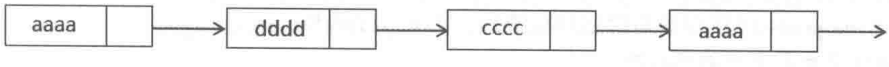


图 1.4　LinkedList 存储示意图

它除了包含 ArrayList 类所包含的方法外，还提供了如表 1-2 所示的一些方法，可以在 LinkedList 的首部或尾部进行插入、删除操作。

表 1-2　LinkedList 类的常用方法

方　法	说　明
void addFirst(Object obj)	将指定元素插入到当前集合的首部
void addLast(Object obj)	将指定元素插入当前集合的尾部
Object getFirst()	获得当前集合的第一个元素
Object getLast()	获得当前集合的最后一个元素
Object removeFirst()	移除并返回当前集合的第一个元素
Object removeLast()	移除并返回当前集合的最后一个元素

示例 3

使用 LinkedList 集合存储新闻标题（包含 ID、名称、创建者），实现获取、添加及删除头条和末条新闻标题信息功能，并遍历集合。

实现步骤：

（1）创建 LinkedList 对象，并添加数据。

（2）添加头条和末尾标题。

（3）获取头条和末条新闻标题信息。

（4）删除头条和末条新闻标题。

关键代码：

```
// 创建多个新闻标题对象
NewTitle car=new NewTitle(1, " 汽车 ", " 管理员 ");
NewTitle medical=new NewTitle(2, " 医学 ", " 管理员 ");
NewTitle fun=new NewTitle(3, " 娱乐 ", " 管理员 ");
NewTitle gym=new NewTitle(4, " 体育 ", " 管理员 ");
// 创建存储新闻标题的集合对象并添加数据
LinkedList newsTitleList=new LinkedList();
newsTitleList.add(car);
newsTitleList.add(medical);
// 添加头条新闻标题和末尾标题
newsTitleList.addFirst(fun);
newsTitleList.addLast(gym);
System.out.println(" 头条和末条新闻已添加 ");
// 获取头条以及最末条新闻标题
NewTitle first=(NewTitle) newsTitleList.getFirst();
System.out.println(" 头条的新闻标题为 :"+first.getTitleName());
NewTitle last=(NewTitle) newsTitleList.getLast();
System.out.println(" 排在最后的新闻标题为 :"+last.getTitleName());
// 删除头条和最末条新闻标题
newsTitleList.removeFirst();
newsTitleList.removeLast();
System.out.println(" 头条和末条新闻已删除 ");
System.out.println(" 遍历所有新闻标题： ");
for(Object obj:newsTitleList){
   NewTitle newTitle=(NewTitle)obj;
   System.out.println(" 新闻标题名称： "+newTitle.getTitleName());
}
```

输出结果如图 1.5 所示。

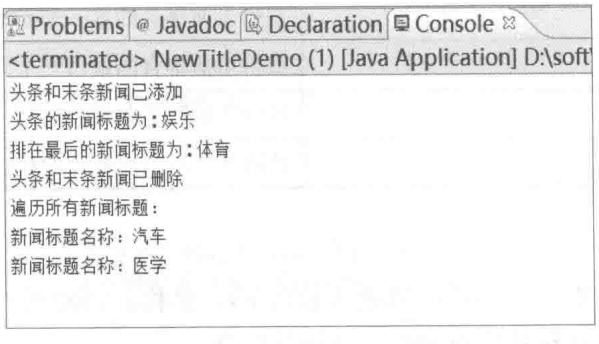


图 1.5　使用 LinkedList 存储并操作新闻标题信息

除了表 1-2 中列出的 LinkedList 类提供的方法外，LinkedList 类和 ArrayList 类所包含的大部分方法是完全一样的，这主要是因为它们都是 List 接口的实现类。由于