# 《大数据基础编程、实验和案例教程（第2版）》

教材官网：
http://dblab.xmu.edu.cn/post/bigdatapractice2/

*温馨提示：编辑幻灯片母版，可以修改每页PPT的厦大校徽和底部文字*

# 第7章 MapReduce基础编程

**（PPT版本号：2020年12月版本）**

扫一扫访问教材官网


**林子雨**

**厦门大学计算机科学系**

**E-mail: ziyulin@xmu.edu.cn**

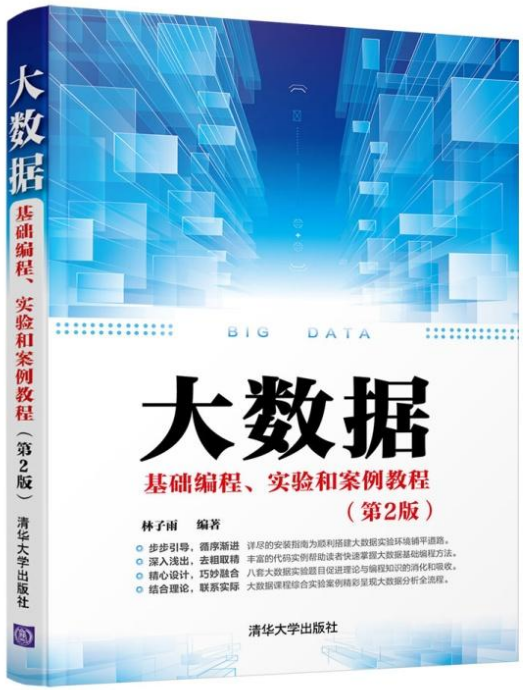
主页：**http://dblab.xmu.edu.cn/linziyu**


# 教材简介

本书是与《大数据技术原理与应用（第3版）》教材配套的唯一指定实验指导书

林子雨编著《大数据基础编程、实验和案例教程（第2版）》
清华大学出版社 ISBN:978-7-302-55977-1 定价：69元，2020年10月第2版
教材官网： http://dblab.xmu.edu.cn/post/bigdatapractice2/

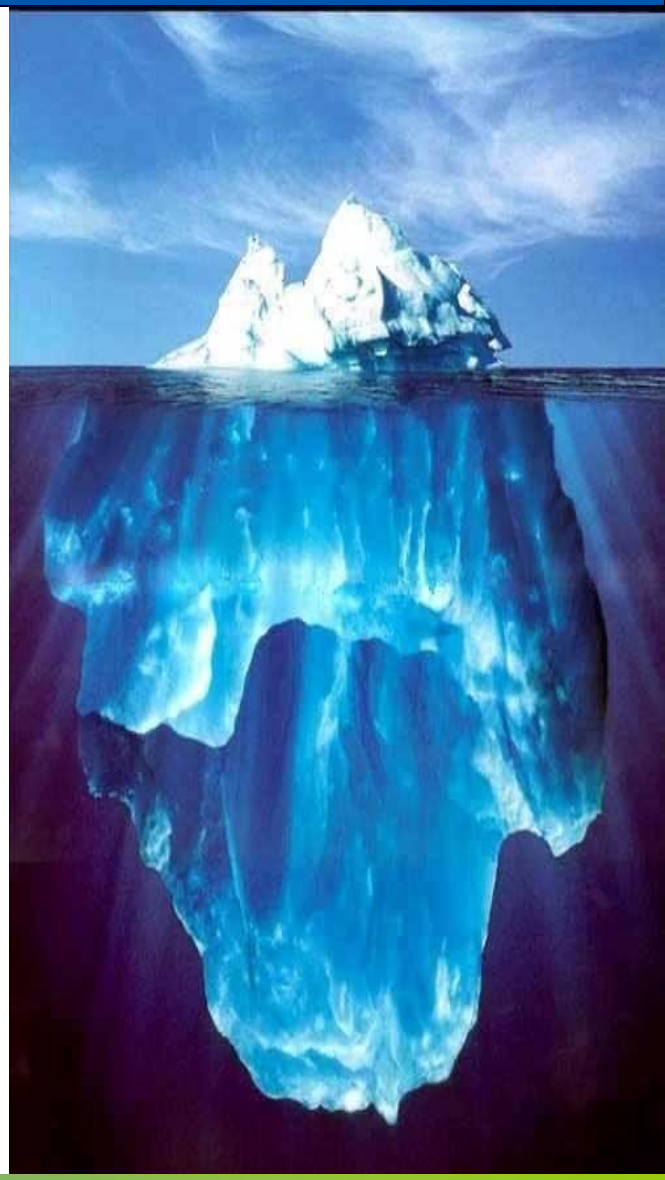
扫一扫访问
教材官网




- 步步引导，循序渐进，详尽的安装指南为顺利搭建大数据实验环境铺平道路
- 深入浅出，去粗取精，丰富的代码实例帮助快速掌握大数据基础编程方法
- 精心设计，巧妙融合，八套大数据实验题目促进理论与编程知识的消化和吸收
- 结合理论，联系实际，大数据课程综合实验案例精彩呈现大数据分析全流程

# 提纲





高校大数据课程
公 共 服 务 平 台

百度搜索厦门大学数据库实验室网站访问平台

# 7.1 词频统计任务要求

在Linux系统本地创建两个文件，即文件wordfile1.txt和wordfile2.txt

文件wordfile1.txt的内容如下：

```
I love Spark
I love Hadoop
```

文件wordfile2.txt的内容如下：

```
Hadoop is good
Spark is fast
```

现在需要设计一个词频统计程序，统计input文件夹下所有文件中每个单词的出现次数

# 7.2 MapReduce程序编写方法

7.2.1 编写Map处理逻辑
7.2.2 编写Reduce处理逻辑

# 7.2.1 编写Map处理逻辑

```
    public static class TokenizerMapper extends Mapper<Object, Text, Text, IntWritable>
{
        private static final IntWritable one = new IntWritable(1);
        private Text word = new Text();
        public TokenizerMapper() {
        }
        public void map(Object key, Text value, Mapper<Object, Text, Text,
IntWritable>.Context context) throws IOException, InterruptedException {
            StringTokenizer itr = new StringTokenizer(value.toString());
            while(itr.hasMoreTokens()) {
                this.word.set(itr.nextToken());
                context.write(this.word, one);
            }
        }
    }
```

# 7.2.2 编写Reduce处理逻辑

```
public static class IntSumReducer extends Reducer<Text, IntWritable, Text, IntWritable> {
        private IntWritable result = new IntWritable();
        public IntSumReducer() {
        }
        public void reduce(Text key, Iterable<IntWritable> values, Reducer<Text, IntWritable,
Text, IntWritable>.Context context) throws IOException, InterruptedException {
            int sum = 0;
            IntWritable val;
            for(Iterator i$ = values.iterator(); i$.hasNext(); sum += val.get()) {
                val = (IntWritable)i$.next();
            }
            this.result.set(sum);
            context.write(key, this.result);
        }
    }
```

# 7.2.3 编写main方法

```
public static void main(String[] args) throws Exception {
        Configuration conf = new Configuration();
        String[] otherArgs = (new GenericOptionsParser(conf, args)).getRemainingArgs();
        if(otherArgs.length < 2) {
            System.err.println("Usage: wordcount <in> [<in>...] <out>");
            System.exit(2);
        }
        Job job = Job.getInstance(conf, "word count");       //设置环境参数
        job.setJarByClass(WordCount.class);                  //设置整个程序的类名
        job.setMapperClass(WordCount.TokenizerMapper.class); //添加Mapper类
        job.setReducerClass(WordCount.IntSumReducer.class);  //添加Reducer类
        job.setOutputKeyClass(Text.class);
                  //设置输出类型
        job.setOutputValueClass(IntWritable.class);          //设置输出类型
        for(int i = 0; i < otherArgs.length - 1; ++i) {
            FileInputFormat.addInputPath(job, new Path(otherArgs[i]));  //设置输入文件
        }
        FileOutputFormat.setOutputPath(job, new Path(otherArgs[otherArgs.length - 1]));//设置输出文
件
        System.exit(job.waitForCompletion(true)?0:1);
    }
```

# 7.2.4 完整的词频统计程序

```
import java.io.IOException;
import java.util.Iterator;
import java.util.StringTokenizer;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.util.GenericOptionsParser;
```

# 7.2.4 完整的词频统计程序

```
public class WordCount {
    public WordCount() {
    }
     public static void main(String[] args) throws Exception {
        Configuration conf = new Configuration();
        String[] otherArgs = (new GenericOptionsParser(conf, args)).getRemainingArgs();
        if(otherArgs.length < 2) {
            System.err.println("Usage: wordcount <in> [<in>...] <out>");
            System.exit(2);
        }
        Job job = Job.getInstance(conf, "word count");
        job.setJarByClass(WordCount.class);
        job.setMapperClass(WordCount.TokenizerMapper.class);
        job.setCombinerClass(WordCount.IntSumReducer.class);
        job.setReducerClass(WordCount.IntSumReducer.class);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);
        for(int i = 0; i < otherArgs.length - 1; ++i) {
            FileInputFormat.addInputPath(job, new Path(otherArgs[i]));
        }
        FileOutputFormat.setOutputPath(job, new Path(otherArgs[otherArgs.length - 1]));
        System.exit(job.waitForCompletion(true)?0:1);
    }
```

# 7.2.4 完整的词频统计程序

```
 public static class TokenizerMapper extends Mapper<Object, Text, Text,
IntWritable> {
        private static final IntWritable one = new IntWritable(1);
        private Text word = new Text();
        public TokenizerMapper() {
        }
        public void map(Object key, Text value, Mapper<Object, Text, Text,
IntWritable>.Context context) throws IOException, InterruptedException
{
            StringTokenizer itr = new StringTokenizer(value.toString());
            while(itr.hasMoreTokens()) {
                this.word.set(itr.nextToken());
                context.write(this.word, one);
            }
        }
    }
```

# 7.2.4 完整的词频统计程序

```
public static class IntSumReducer extends Reducer<Text, IntWritable,
Text, IntWritable> {
        private IntWritable result = new IntWritable();
        public IntSumReducer() {
        }
        public void reduce(Text key, Iterable<IntWritable> values,
Reducer<Text, IntWritable, Text, IntWritable>.Context context) throws
IOException, InterruptedException {
            int sum = 0;
            IntWritable val;
            for(Iterator i$ = values.iterator(); i$.hasNext(); sum += val.get()) {
                val = (IntWritable)i$.next();
            }
            this.result.set(sum);
            context.write(key, this.result);
        }
    }
}
```

# 7.3 编译打包程序

7.3.1 使用命令行编译打包词频统计程序
7.3.2 使用Eclipse编译运行词频统计程序

# 7.3.1 使用命令行编译打包词频统计程序

首先，请在Linux系统中打开一个终端，把Hadoop的安装目录设置为当前工作目录，命令如下：

```
$ cd /usr/local/hadoop
```

然后，执行如下命令，让javac编译程序可以找到Hadoop相关的JAR包：

```
$export
CLASSPATH="/usr/local/hadoop/share/hadoop/common/hadoop-
common-
3.1.3.jar:/usr/local/hadoop/share/hadoop/mapreduce/hadoop-
mapreduce-client-core-
3.1.3.jar:/usr/local/hadoop/share/hadoop/common/lib/commons-cli-
1.2.jar:$CLASSPATH"
```

# 7.3.1 使用命令行编译打包词频统计程序

接下来，就可以执行javac命令来编译程序（这里假设WordCount.java文件被放在了“/usr/local/hadoop”目录下）：

```
$ javac WordCount.java
```

编译之后，在文件夹下可以发现有3个“.class”文件，这是Java的可执行文件。此时，我们需要将它们打包并命名为WordCount.jar，命令如下：

```
$ jar -cvf WordCount.jar *.class
```

到这里，我们就得到像Hadoop自带实例一样的jar包了，可以运行得到结果。在运行程序之前，需要使用命令start-dfs.sh启动Hadoop。启动Hadoop之后，我们可以运行程序，命令如下：

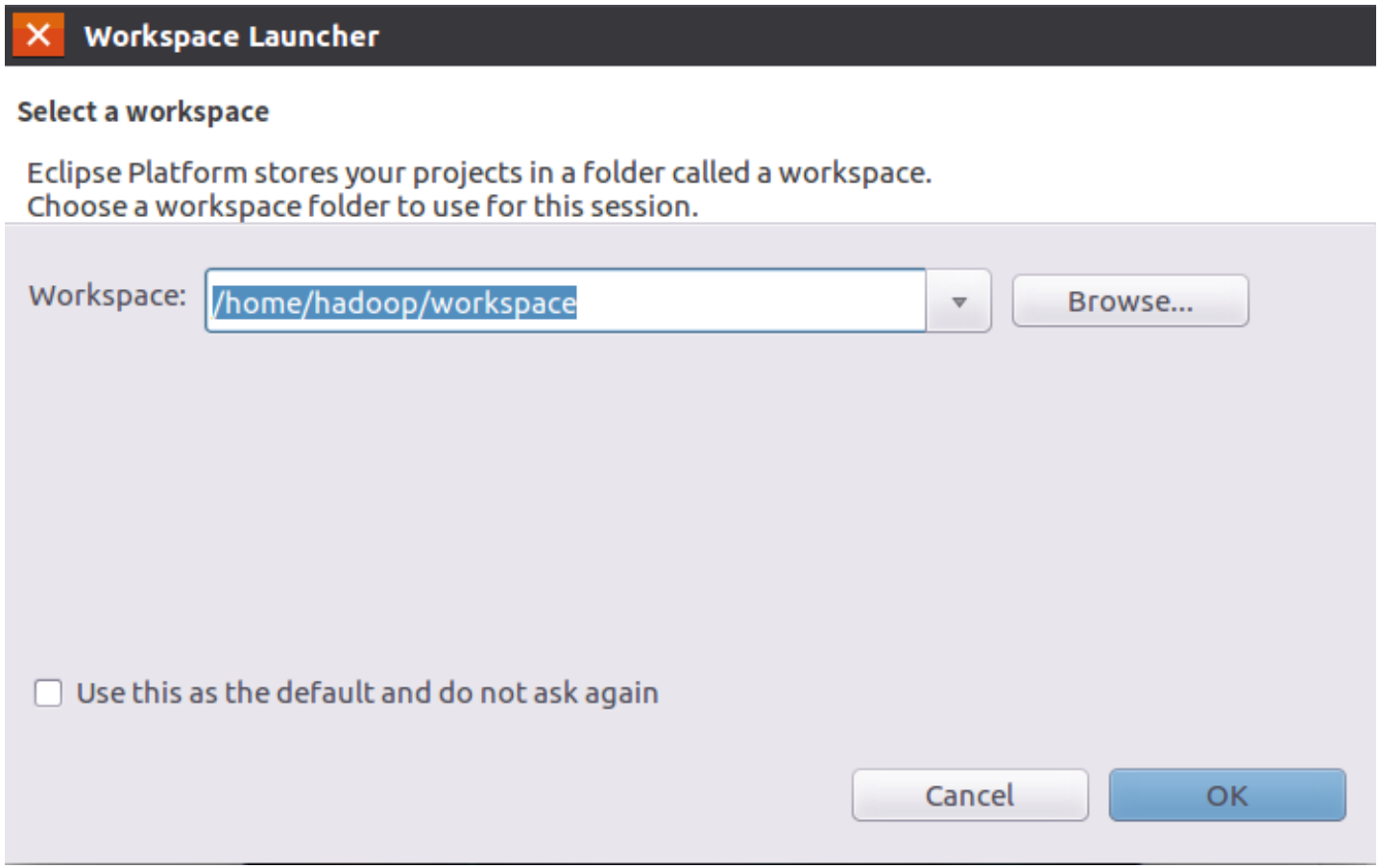
```
$ ./bin/hadoop jar WordCount.jar WordCount input output
```

最后，可以运行下面命令查看结果：

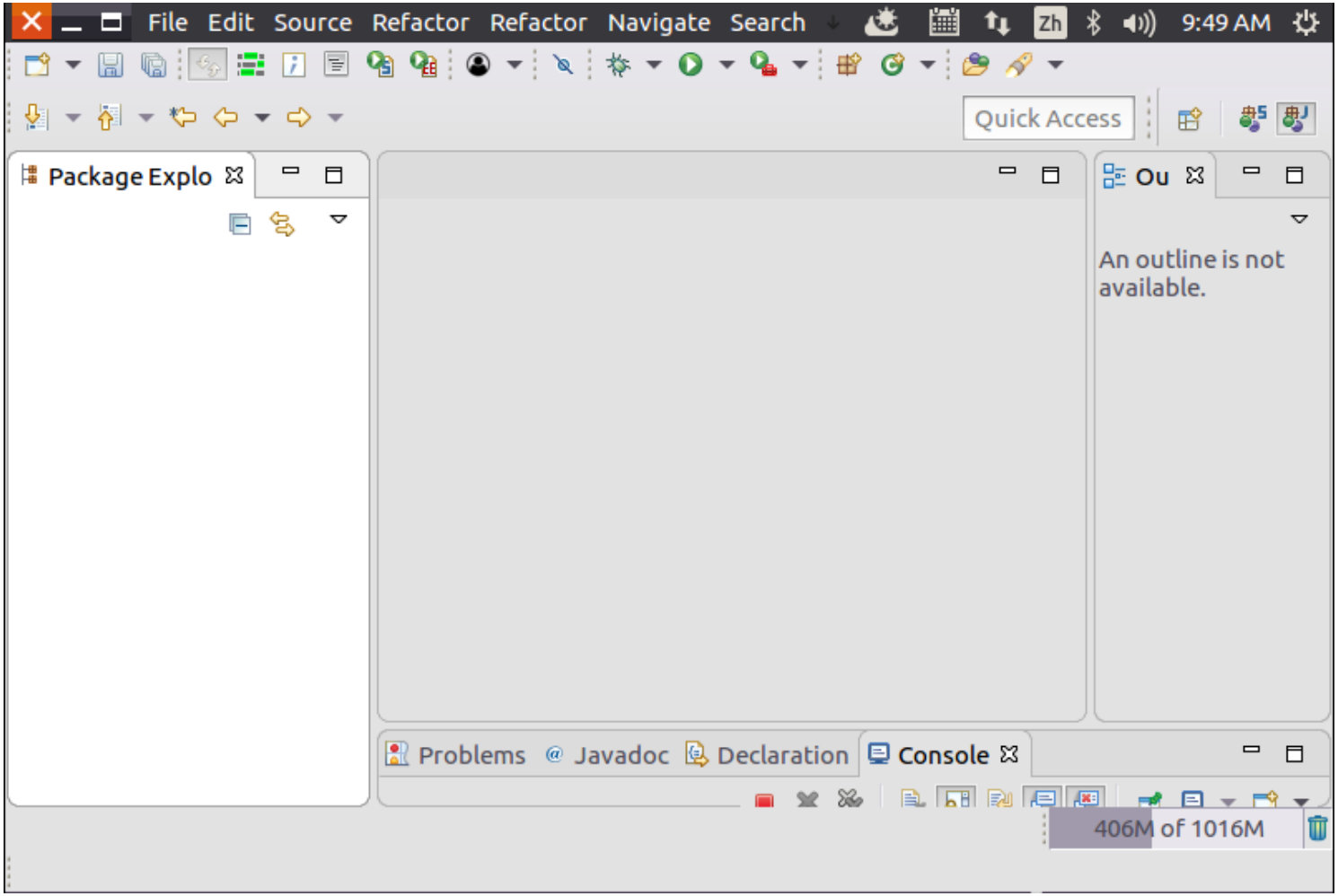
```
$ ./bin/hadoop fs -cat output/*
```

# 7.3.2 使用Eclipse编译运行词频统计程序

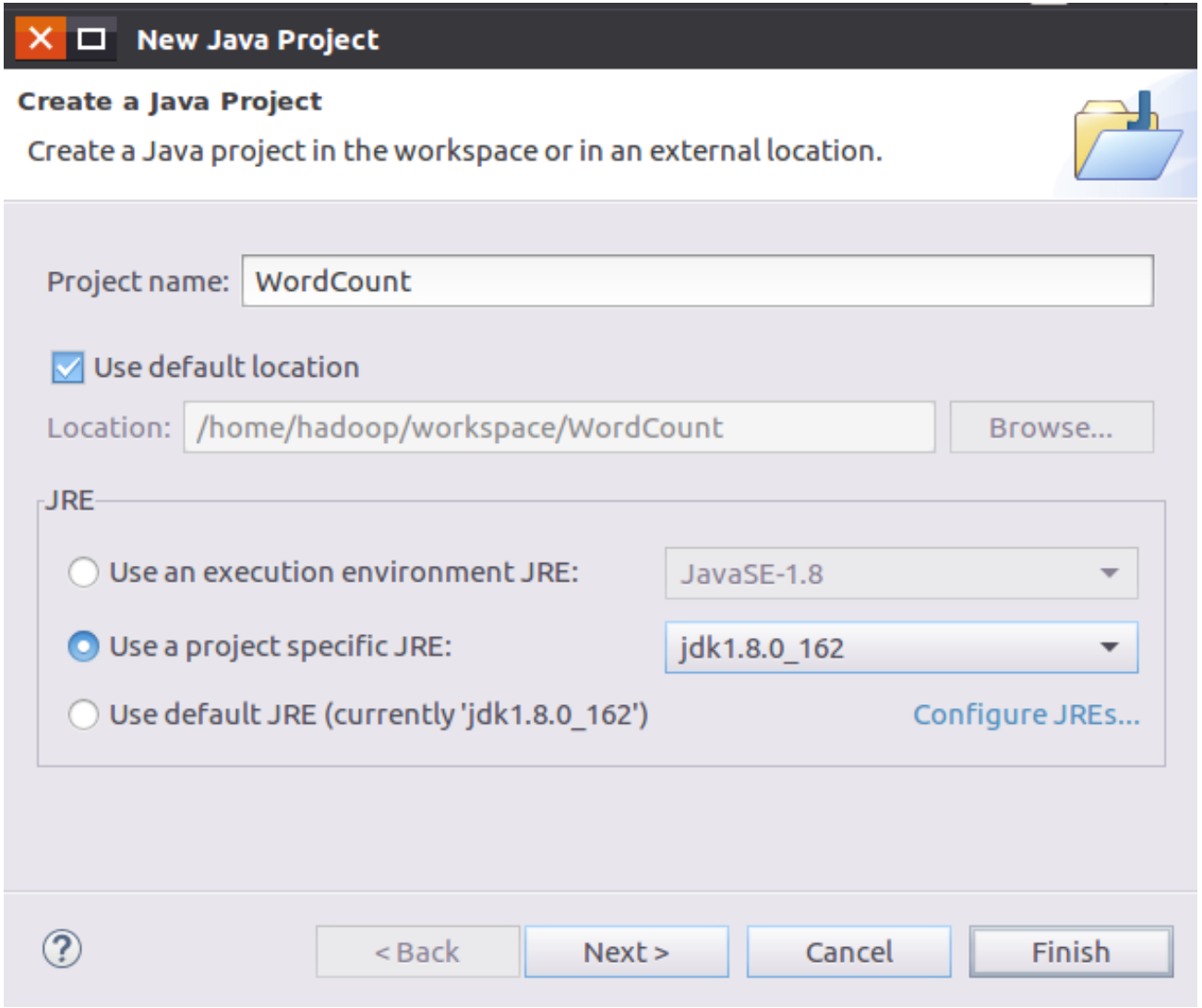
**1. 在Eclipse中创建项目**



# 7.3.2 使用Eclipse编译运行词频统计程序

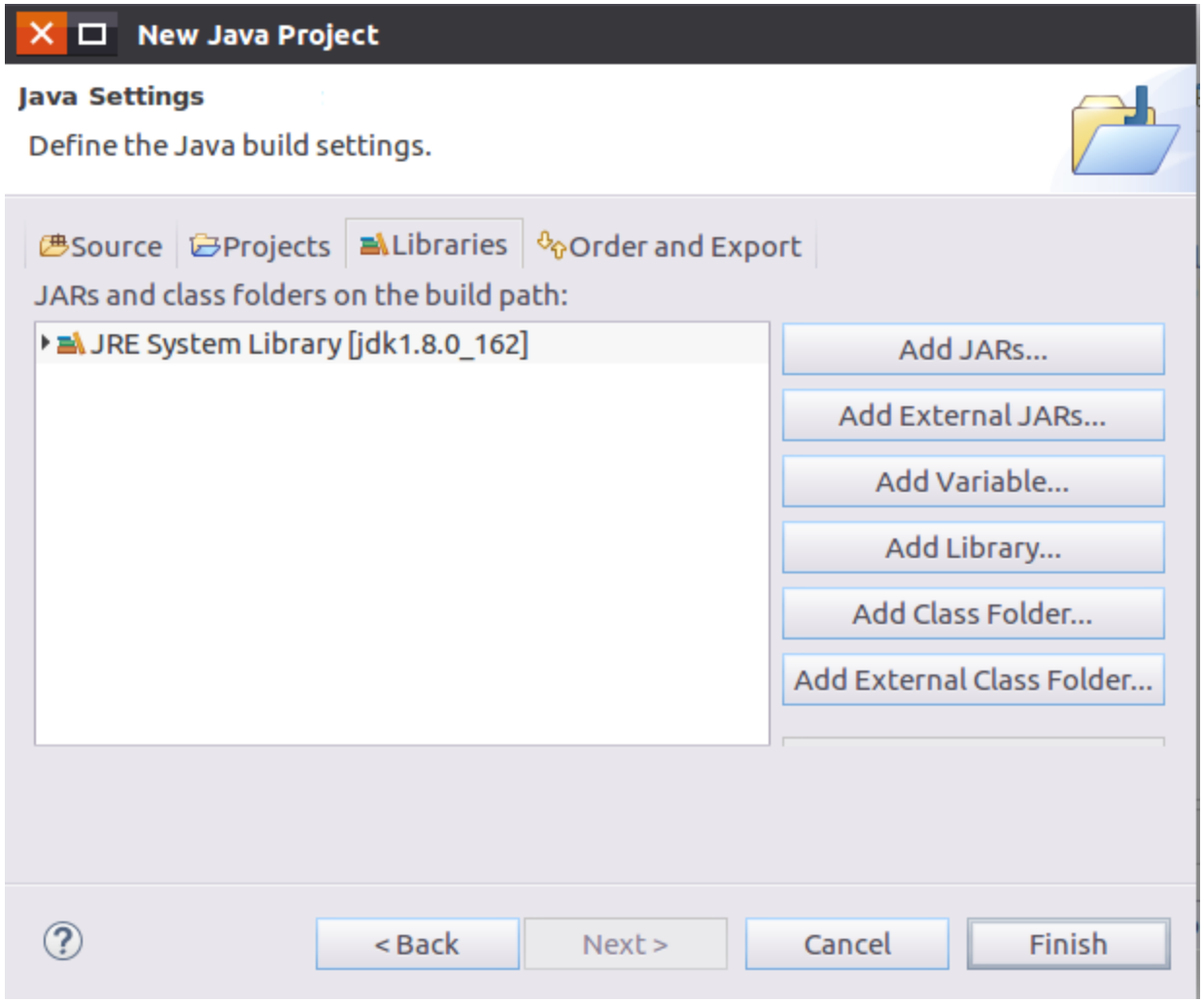


# 7.3.2 使用Eclipse编译运行词频统计程序

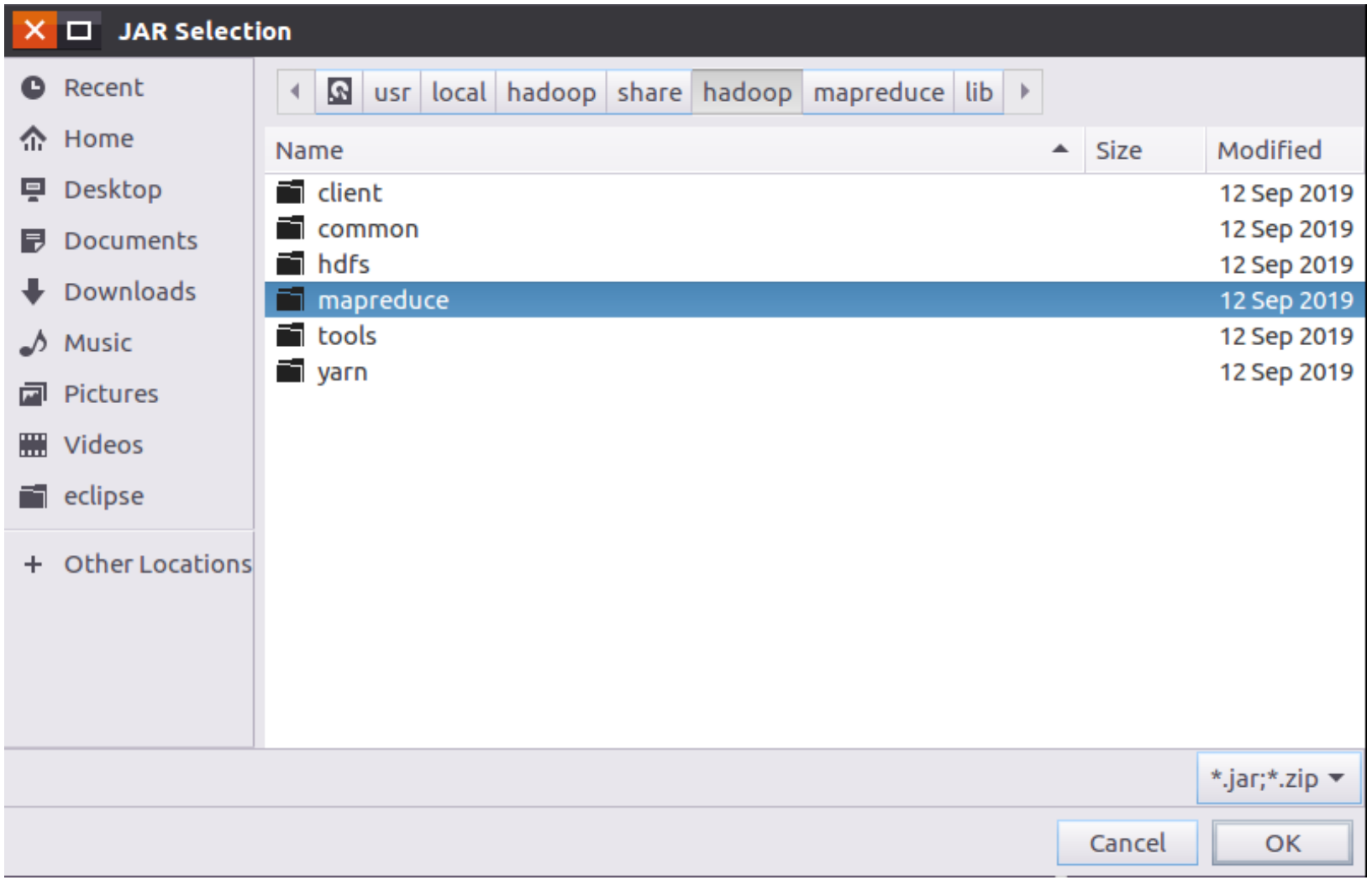


# 7.3.2 使用Eclipse编译运行词频统计程序

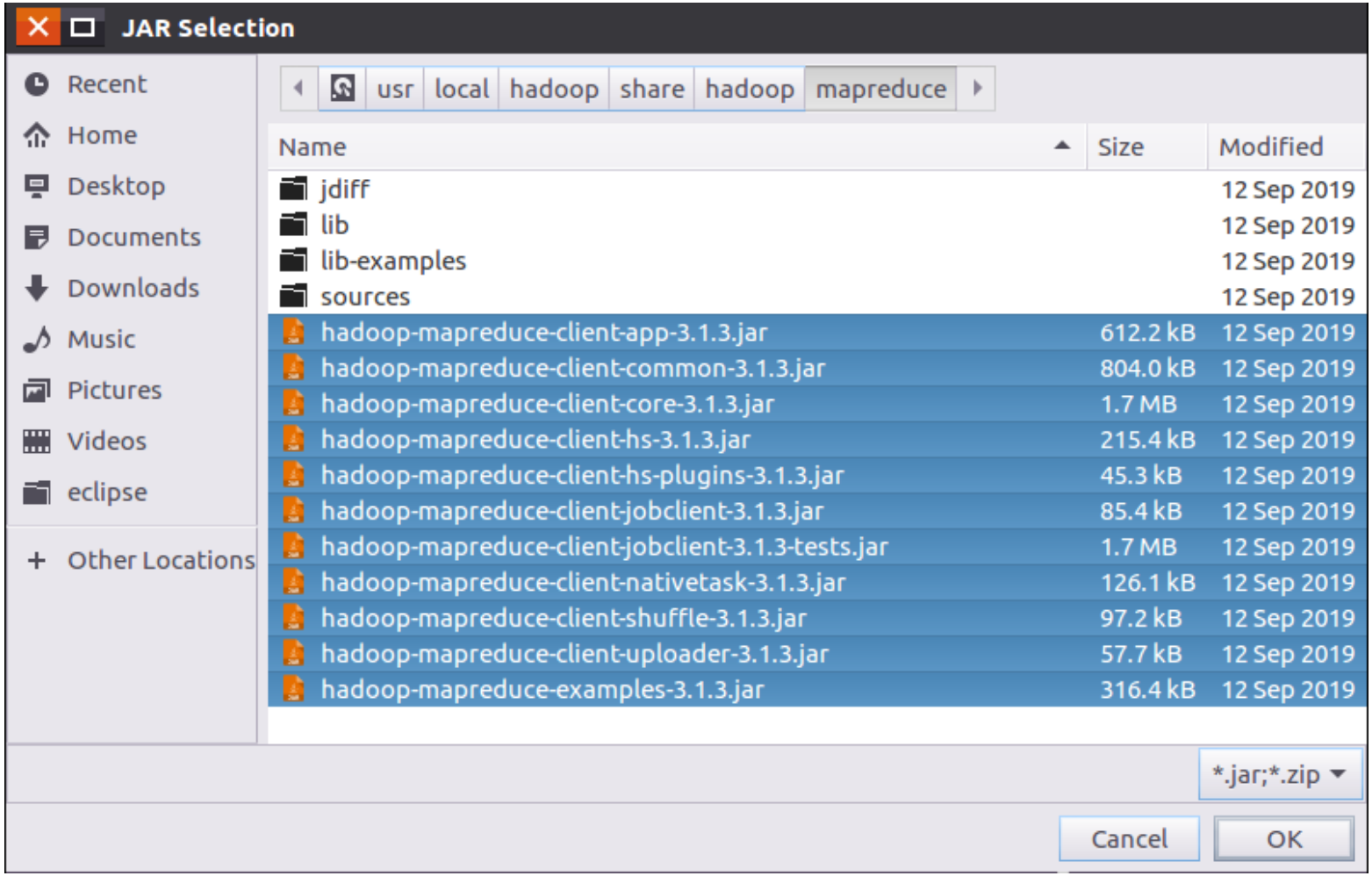
**2. 为项目添加需要用到的JAR包**



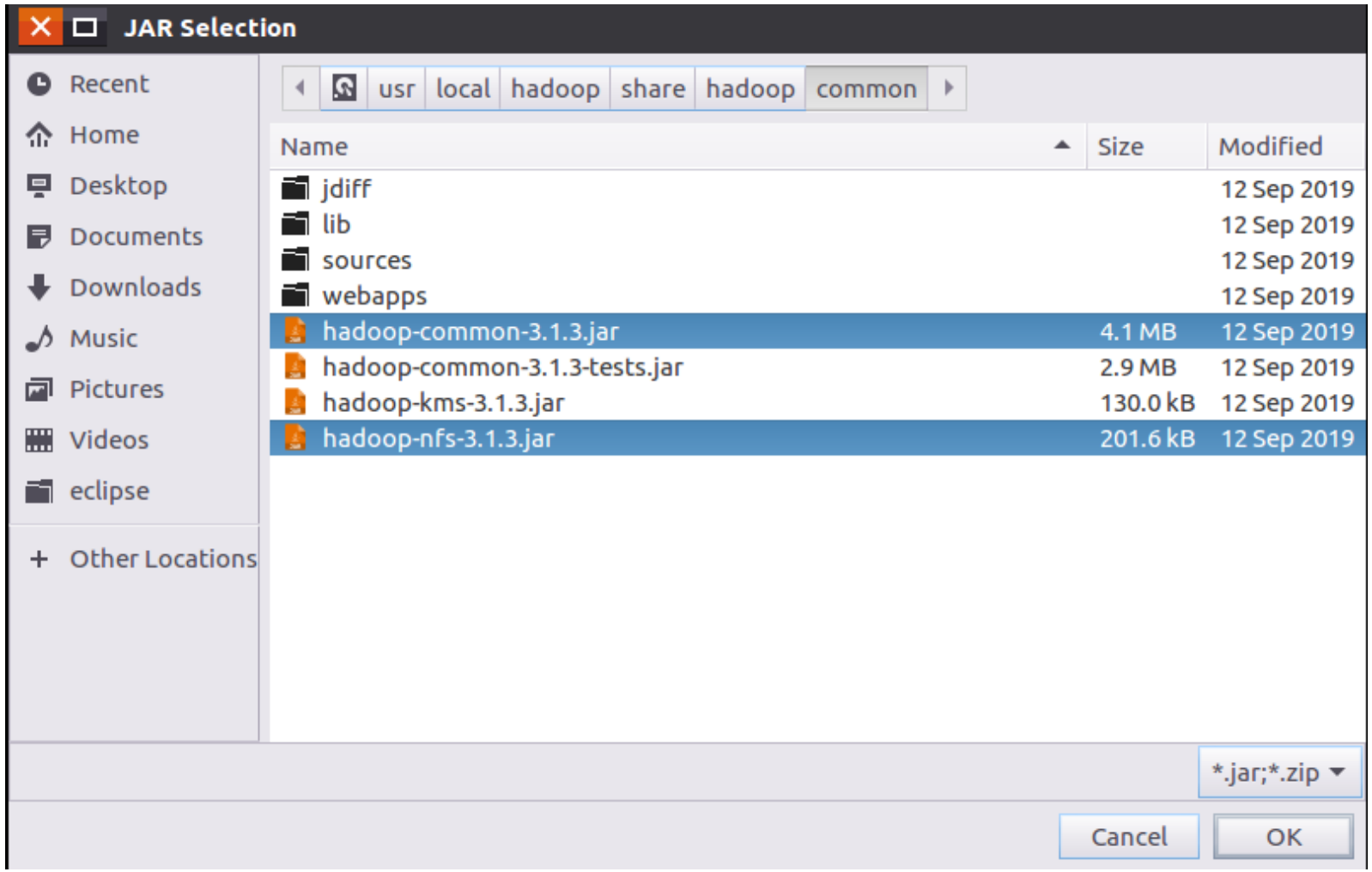
# 7.3.2 使用Eclipse编译运行词频统计程序



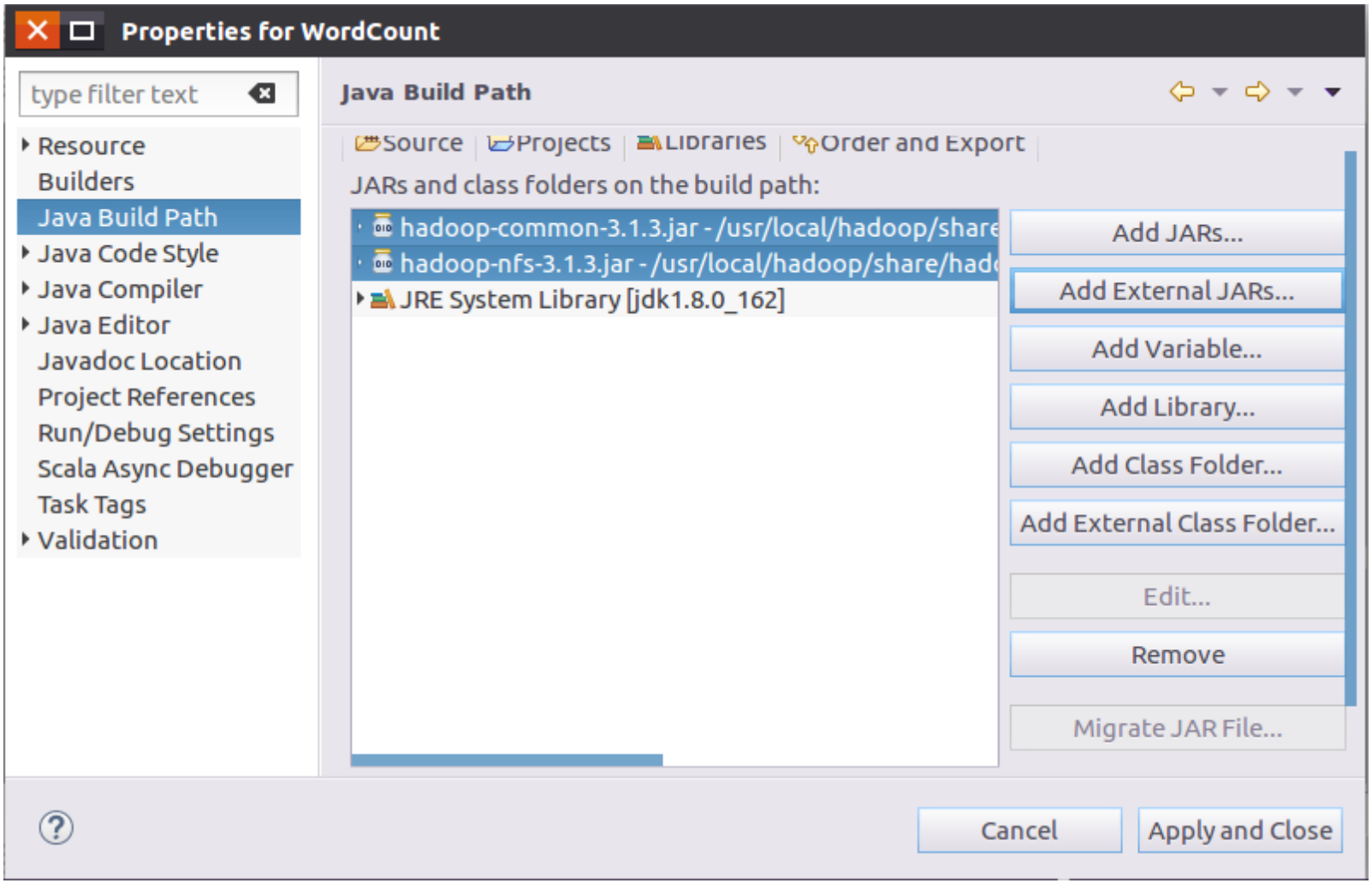
# 7.3.2 使用Eclipse编译运行词频统计程序



# 7.3.2 使用Eclipse编译运行词频统计程序

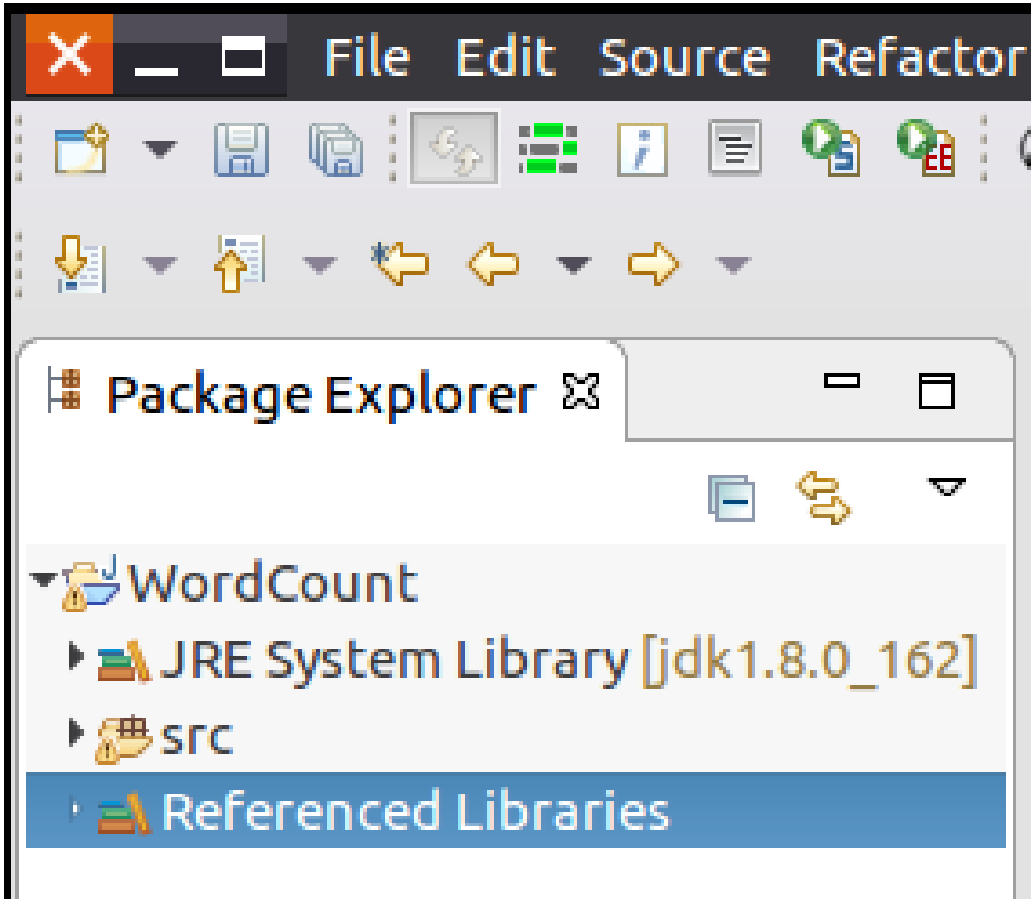


# 7.3.2 使用Eclipse编译运行词频统计程序

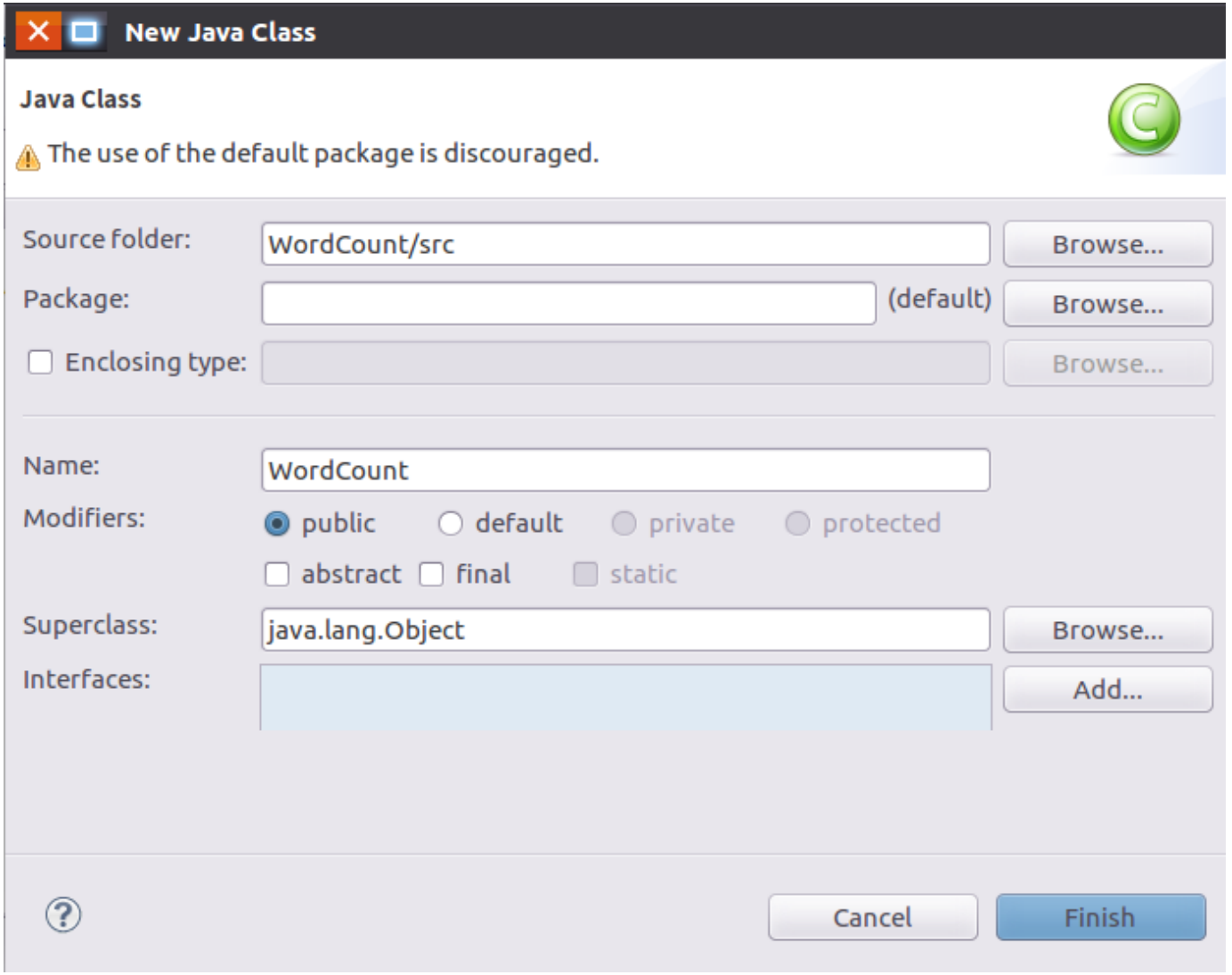


# 7.3.2 使用Eclipse编译运行词频统计程序

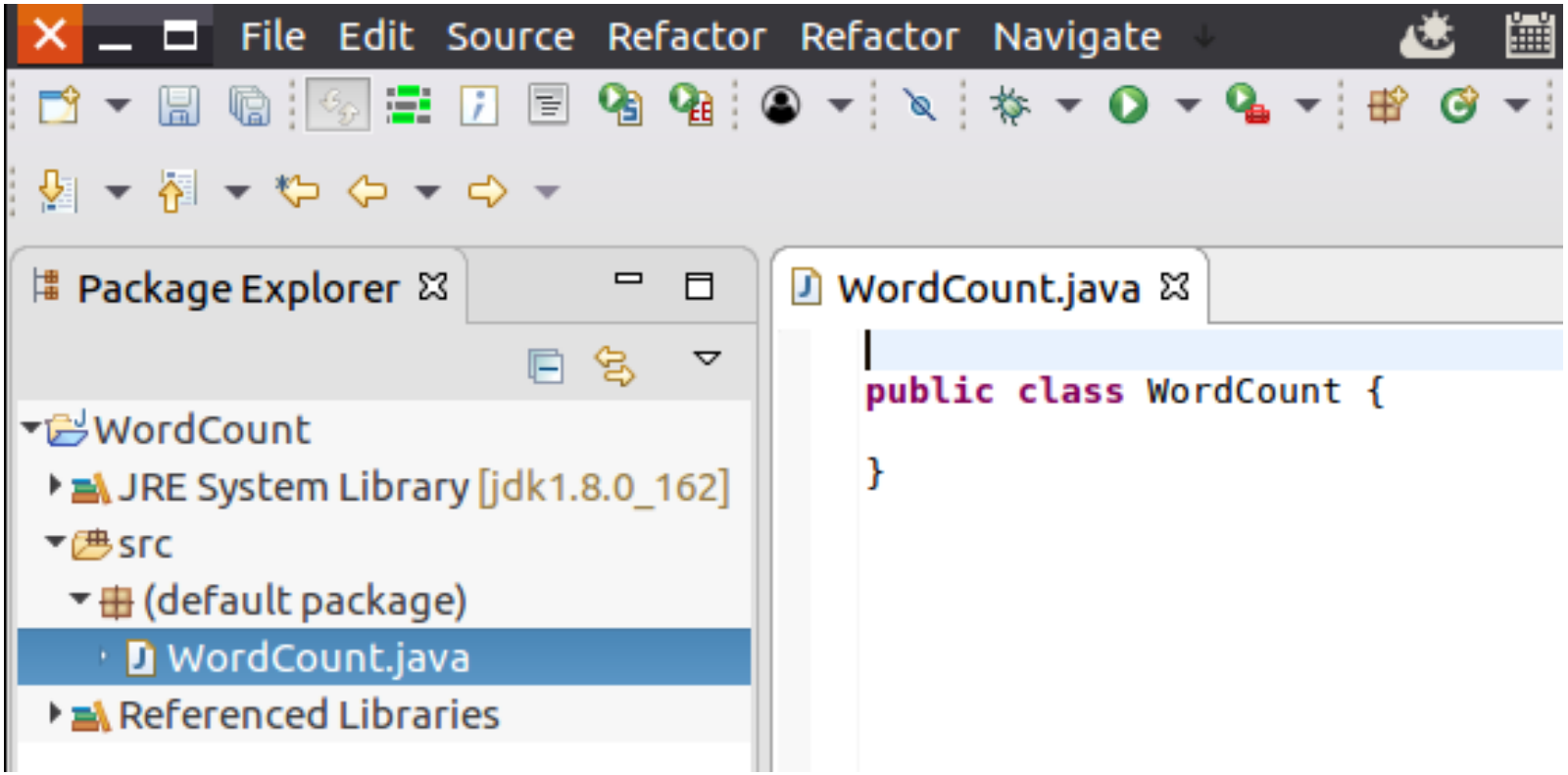
**3. 编写Java应用程序**



# 7.3.2 使用Eclipse编译运行词频统计程序

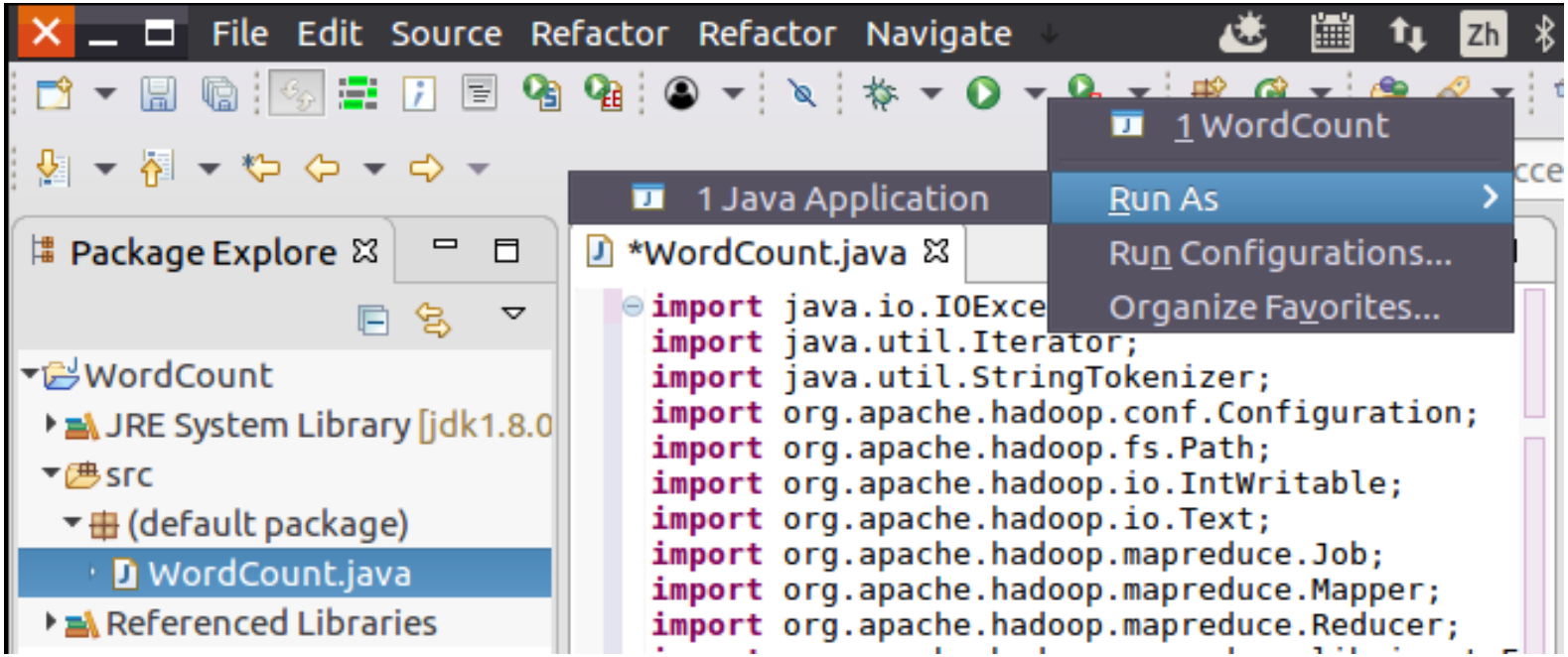


# 7.3.2 使用Eclipse编译运行词频统计程序

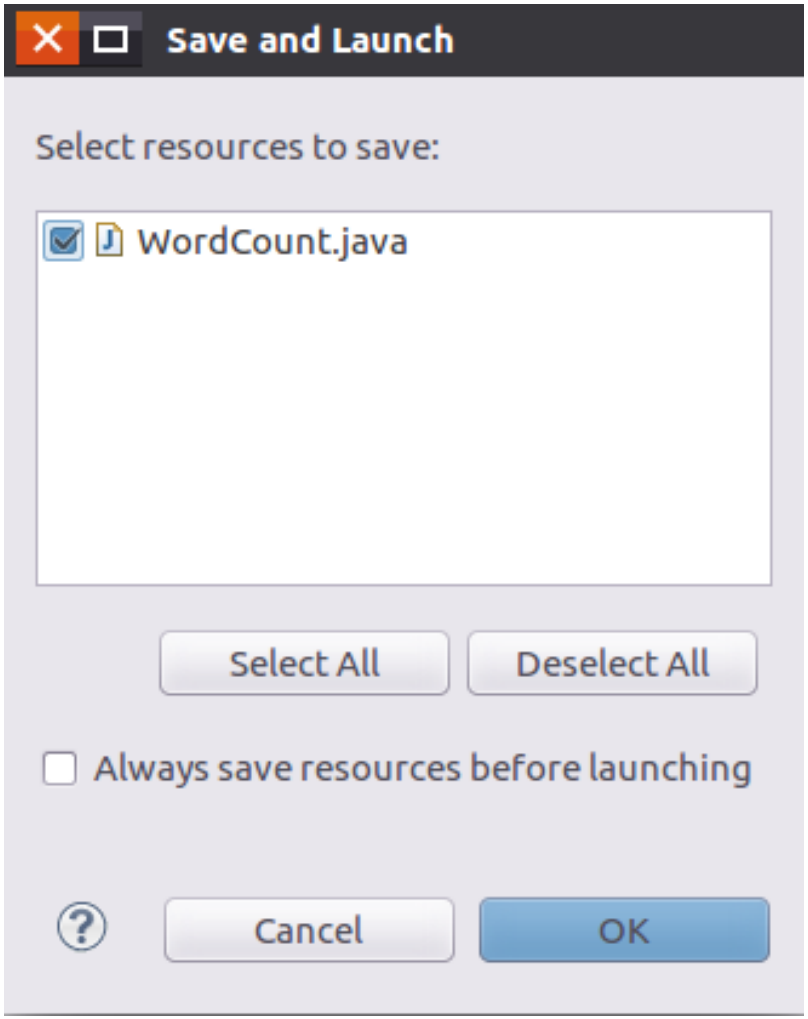


# 7.3.2 使用Eclipse编译运行词频统计程序

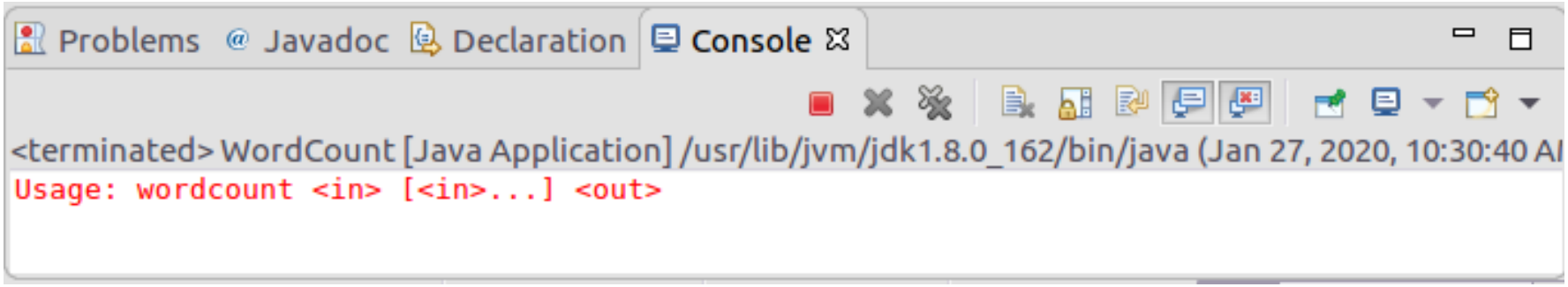
**4. 编译打包程序**



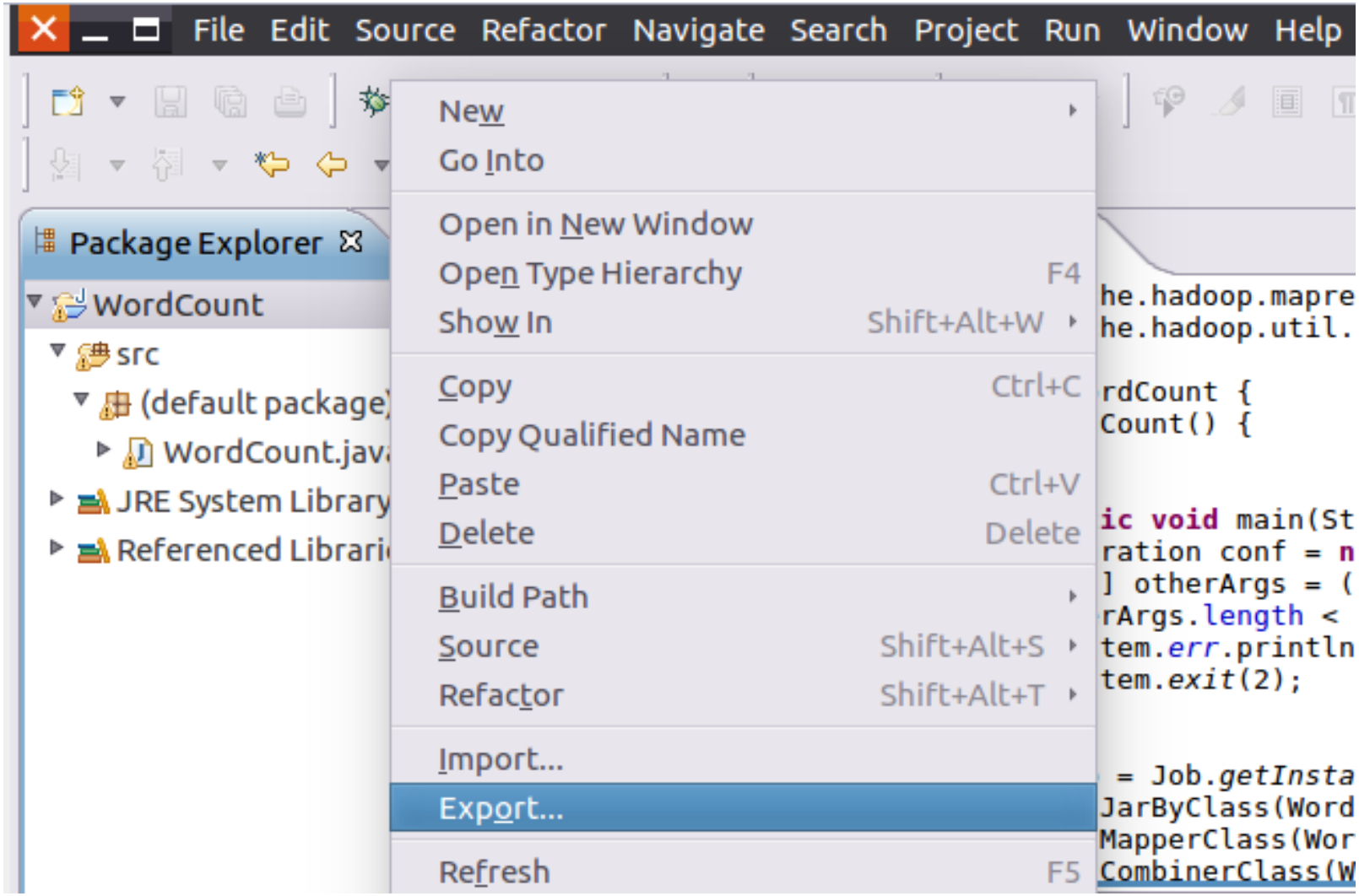
# 7.3.2 使用Eclipse编译运行词频统计程序



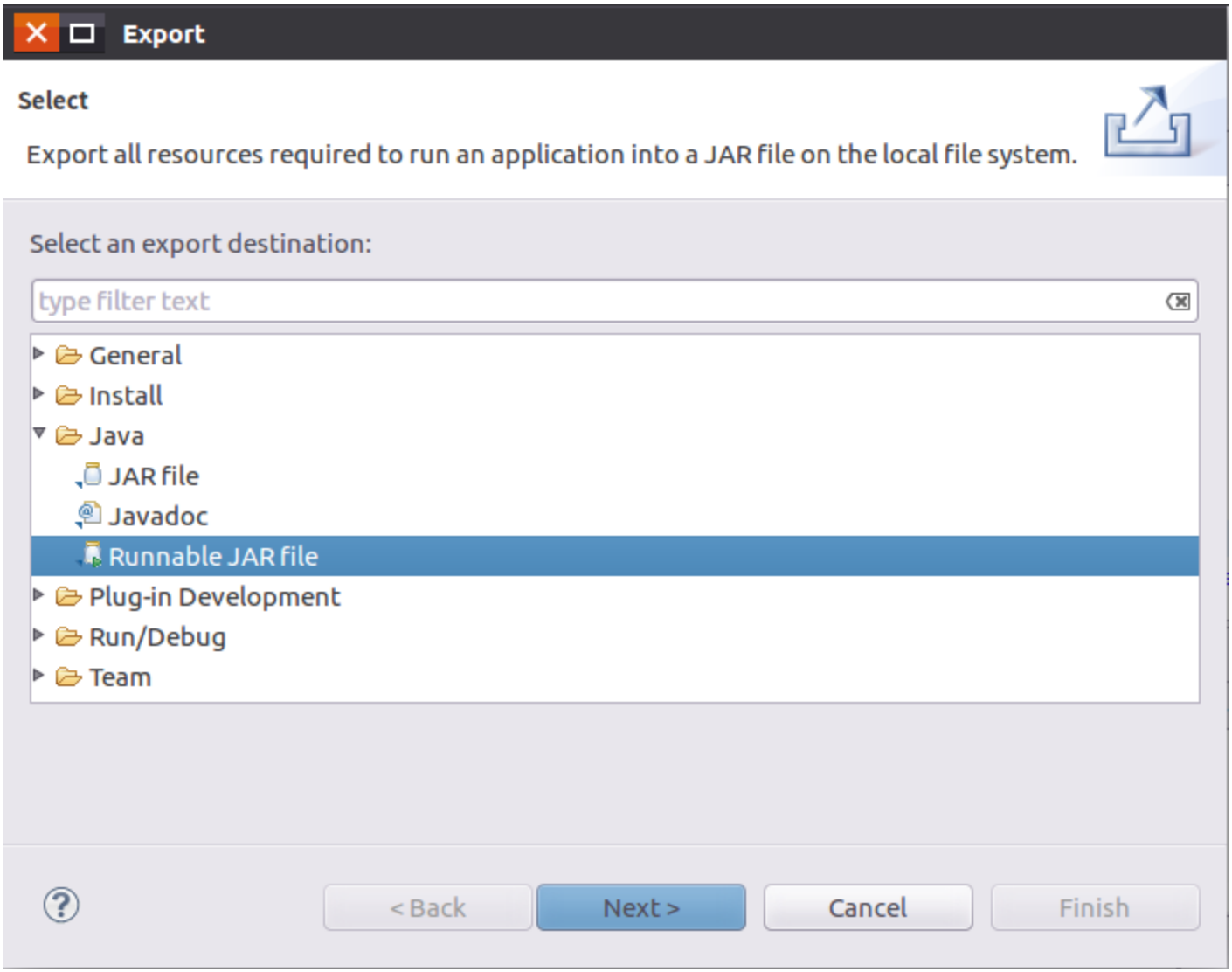
# 7.3.2 使用Eclipse编译运行词频统计程序



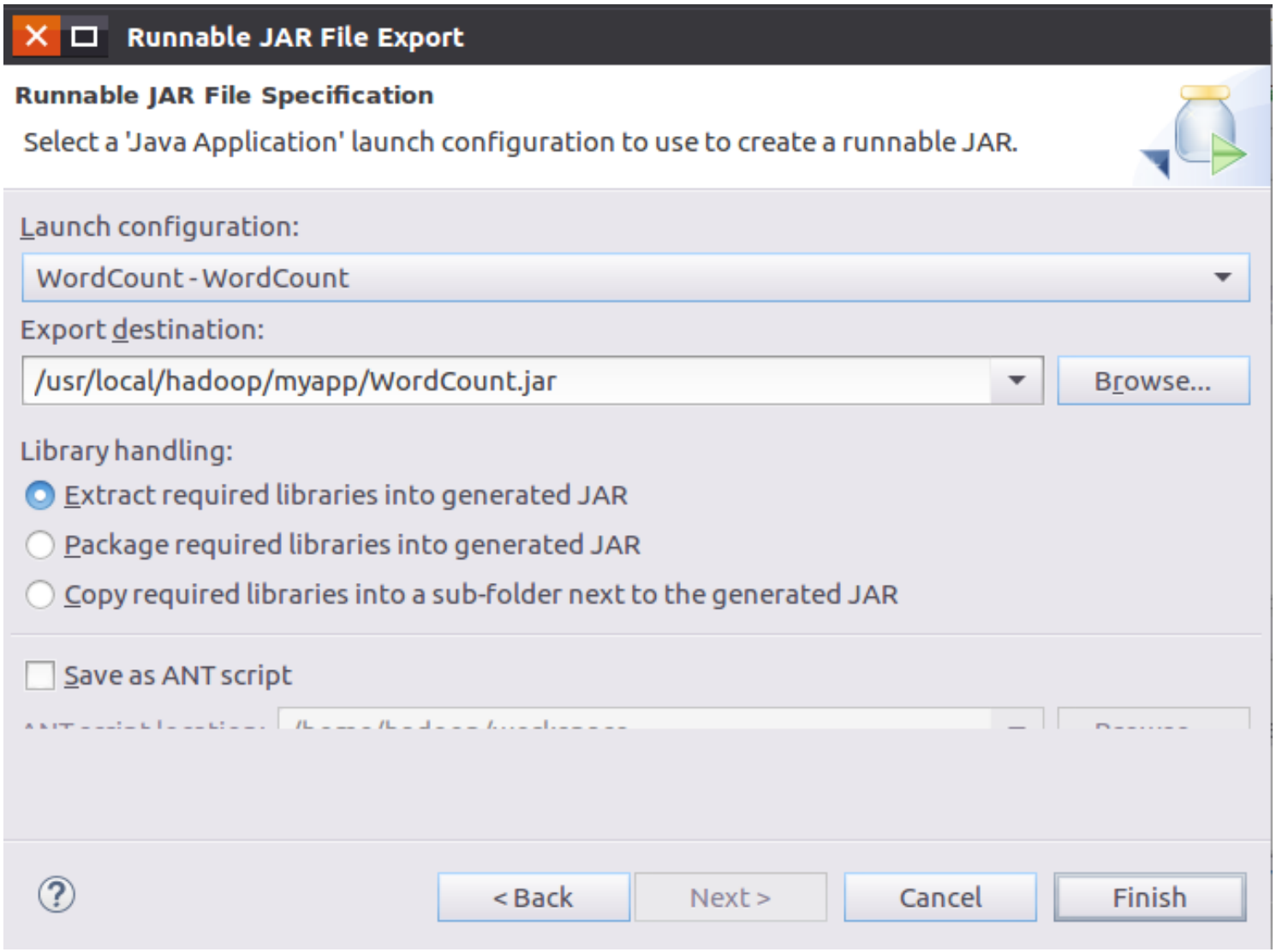
# 7.3.2 使用Eclipse编译运行词频统计程序



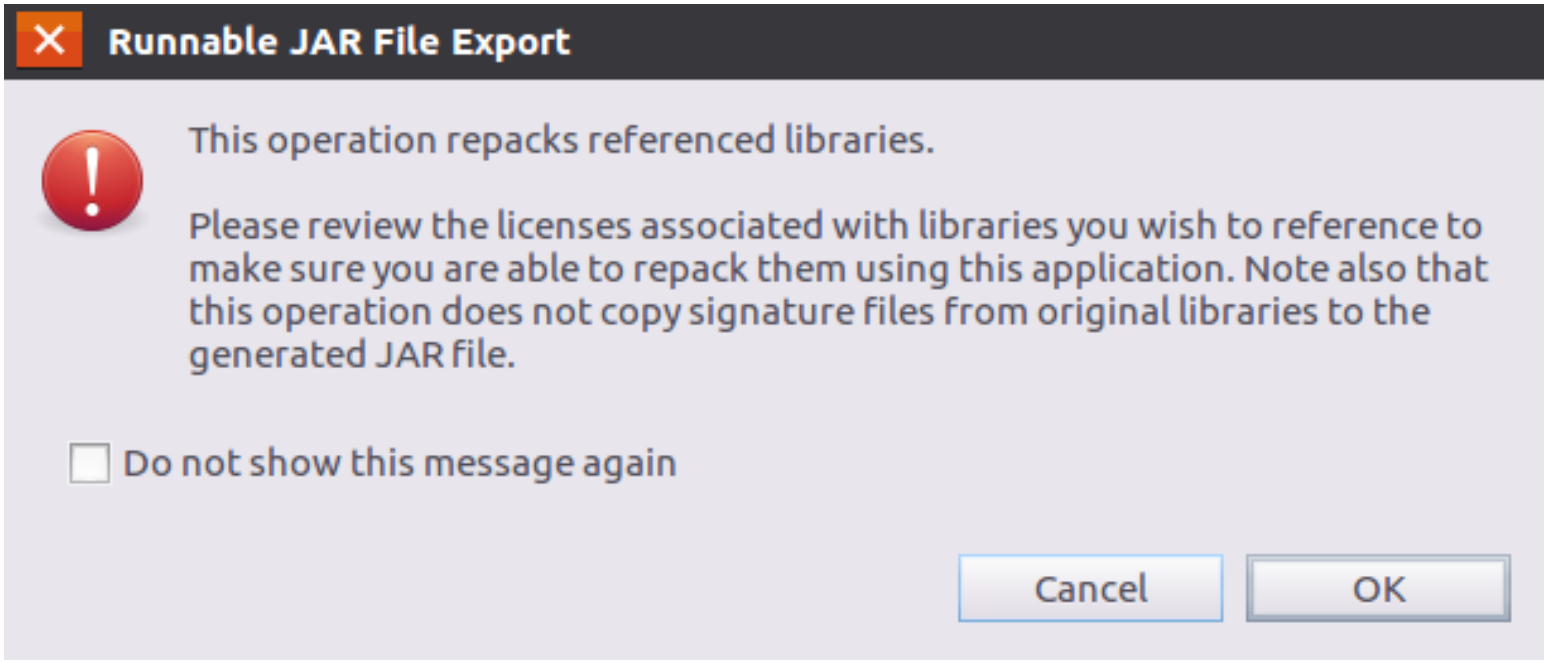
# 7.3.2 使用Eclipse编译运行词频统计程序



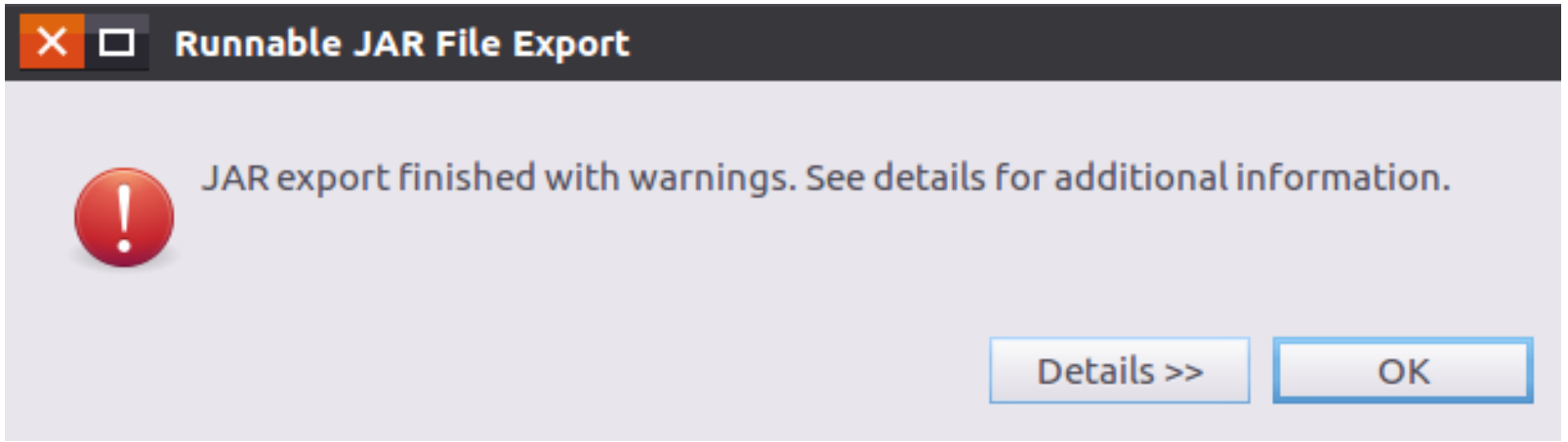
# 7.3.2 使用Eclipse编译运行词频统计程序



# 7.3.2 使用Eclipse编译运行词频统计程序



# 7.3.2 使用Eclipse编译运行词频统计程序




可以到Linux系统中查看一下生成的WordCount.jar文件，可以在Linux的终端中执行如下命令：

```
$ cd /usr/local/hadoop/myapp
$ ls
```

# 7.4 运行程序

在运行程序之前，需要启动Hadoop，命令如下：

```
$ cd /usr/local/hadoop
$ ./sbin/start-dfs.sh
```

在启动Hadoop之后，需要首先删除HDFS中与当前Linux用户hadoop对应的input和output目录（即HDFS中的“/user/hadoop/input”和“/user/hadoop/output”目录），这样确保后面程序运行不会出现问题，具体命令如下：

```
$ cd /usr/local/hadoop
$ ./bin/hdfs dfs -rm -r input
$ ./bin/hdfs dfs -rm -r output
```

# 7.4 运行程序

然后，再在HDFS中新建与当前Linux用户hadoop对应的input目录，即“/user/hadoop/input”目录，具体命令如下：

```
$ cd /usr/local/hadoop
$ ./bin/hdfs dfs -mkdir input
```

然后，把之前在第7.1节中在Linux本地文件系统中新建的两个文件wordfile1.txt和wordfile2.txt（假设这两个文件位于“/usr/local/hadoop”目录下，并且里面包含了一些英文语句），上传到HDFS中的“/user/hadoop/input”目录下，命令如下：

```
$ cd /usr/local/hadoop
$ ./bin/hdfs dfs -put ./wordfile1.txt input
$ ./bin/hdfs dfs -put ./wordfile2.txt input
```

# 7.4 运行程序

如果HDFS中已经存在目录“/user/hadoop/output”，则使用如下命令删除该目录：

```
$ cd /usr/local/hadoop
$ ./bin/hdfs dfs -rm -r /user/hadoop/output
```

现在，就可以在Linux系统中，使用hadoop jar命令运行程序，命令如下：

```
$ cd /usr/local/hadoop
$ ./bin/hadoop jar ./myapp/WordCount.jar input output
```

词频统计结果已经被写入了HDFS的“/user/hadoop/output”目录中，可以执行如下命令查看词频统计结果：

```
$ cd /usr/local/hadoop
$ ./bin/hdfs dfs -cat output/*
```

# 7.5 本章小结

本章详细演示了如何编写MapReduce程序实现词频统计功能。在编写MapReduce程序之前，需要先判断目标任务是否可以采用MapReduce编程。MapReduce会把一个大的文件切分成多小片段进行分布式并行处理，最终对不同片段的处理结果进行汇总。很显然，词频统计任务是符合这个要求的，因此，可以采用MapReduce编写程序。

本章详细介绍了MapReduce程序的具体编写方法，包括编写Map处理逻辑、Reduce处理逻辑、main方法等。最后，演示了如何使用Eclipse编译运行Java应用程序。通过本章的学习，可以形成对MapReduce编程方法的基本认识。如果要深入了解如何把各种任务转换成MapReduce程序，建议继续学习相关的进阶书籍。

# 附录A：主讲教师林子雨简介

**主讲教师：林子雨**


单位：厦门大学计算机科学系
E-mail: ziyulin@xmu.edu.cn
个人网页：http://dblab.xmu.edu.cn/post/linziyu
数据库实验室网站：http://dblab.xmu.edu.cn


扫一扫访问个人主页

林子雨，男，1978年出生，博士（毕业于北京大学），全国高校知名大数据教师，现为厦门大学计算机科学系副教授，曾任厦门大学信息科学与技术学院院长助理、晋江市发展和改革局副局长。中国计算机学会数据库专业委员会委员，中国计算机学会信息系统专业委员会委员。国内高校首个“数字教师”提出者和建设者，厦门大学数据库实验室负责人，厦门大学云计算与大数据研究中心主要建设者和骨干成员，2013年度、2017年度和2020年度厦门大学教学类奖教金获得者，荣获2019年福建省精品在线开放课程、2018年厦门大学高等教育成果特等奖、2018年福建省高等教育教学成果二等奖、2018年国家精品在线开放课程。主要研究方向为数据库、数据仓库、数据挖掘、大数据、云计算和物联网，并以第一作者身份在《软件学报》《计算机学报》和《计算机研究与发展》等国家重点期刊以及国际学术会议上发表多篇学术论文。作为项目负责人主持的科研项目包括1项国家自然科学青年基金项目(No.61303004)、1项福建省自然科学青年基金项目(No.2013J05099)和1项中央高校基本科研业务费项目(No.2011121049)，主持的教改课题包括1项2016年福建省教改课题和1项2016年教育部产学协作育人项目，同时，作为课题负责人完成了国家发改委城市信息化重大课题、国家物联网重大应用示范工程区域试点泉州市工作方案、2015泉州市互联网经济调研等课题。中国高校首个“数字教师”提出者和建设者，2009年至今，“数字教师”大平台累计向网络免费发布超过1000万字高价值的研究和教学资料，累计网络访问量超过1000万次。打造了中国高校大数据教学知名品牌，编著出版了中国高校第一本系统介绍大数据知识的专业教材《大数据技术原理与应用》，并成为京东、当当网等网店畅销书籍；建设了国内高校首个大数据课程公共服务平台，为教师教学和学生学习大数据课程提供全方位、一站式服务，年访问量超过200万次，累计访问量超过1000万次。

# 附录B：大数据学习路线图


大数据学习路线图访问地址：http://dblab.xmu.edu.cn/post/10164/

# 附录C：林子雨大数据系列教材


了解全部教材信息：http://dblab.xmu.edu.cn/post/bigdatabook/

# 附录D：《大数据导论（通识课版）》教材

## 开设全校公共选修课的优质教材


本课程旨在实现以下几个培养目标：

- 引导学生步入大数据时代，积极投身大数据的变革浪潮之中
- 了解大数据概念，培养大数据思维，养成数据安全意识
- 认识大数据伦理，努力使自己的行为符合大数据伦理规范要求
- 熟悉大数据应用，探寻大数据与自己专业的应用结合点
- 激发学生基于大数据的创新创业热情

高等教育出版社 ISBN:978-7-04-053577-8 定价：32元

教材官网：http://dblab.xmu.edu.cn/post/bigdataintroduction/

# 附录E：《大数据导论》教材

- 林子雨 编著《大数据导论》
- 人民邮电出版社，2020年9月第1版
- ISBN:978-7-115-54446-9 定价：49.80元

教材官网：http://dblab.xmu.edu.cn/post/bigdata-introduction/


**开设大数据专业导论课的优质教材**

扫一扫访问教材官网

# 附录F：《大数据技术原理与应用》教材

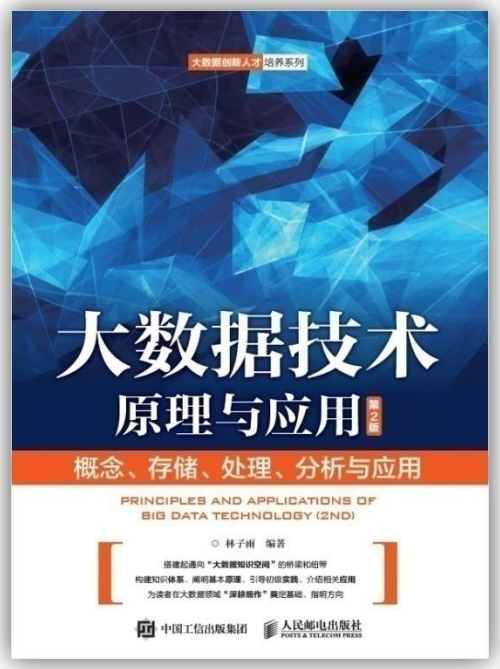
《大数据技术原理与应用——概念、存储、处理、分析与应用（第2版）》，由厦门大学计算机科学系林子雨博士编著，是国内高校第一本系统介绍大数据知识的专业教材。人民邮电出版社 ISBN:978-7-115-44330-4 定价：49.80元

全书共有15章，系统地论述了大数据的基本概念、大数据处理架构Hadoop、分布式文件系统HDFS、分布式数据 库HBase、NoSQL数据库、云数据库、分布式并行编程模型MapReduce、Spark、流计算、图计算、数据可视化以及大数据在互联网、生物医学和物流等各个领域的应用。在Hadoop、HDFS、HBase和MapReduce等重要章节，安排了入门级的实践操作，让读者更好地学习和掌握大数据关键技术。

本书可以作为高等院校计算机专业、信息管理等相关专业的大数据课程教材，也可供相关技术人员参考、学习、培训之用。

欢迎访问《大数据技术原理与应用——概念、存储、处理、分析与应用》教材官方网站：
http://dblab.xmu.edu.cn/post/bigdata

扫一扫访问教材官网



# 附录G：《大数据基础编程、实验和案例教程（第2版）》

本书是与《大数据技术原理与应用（第3版）》教材配套的唯一指定实验指导书

大数据教材


1+1黄金组合

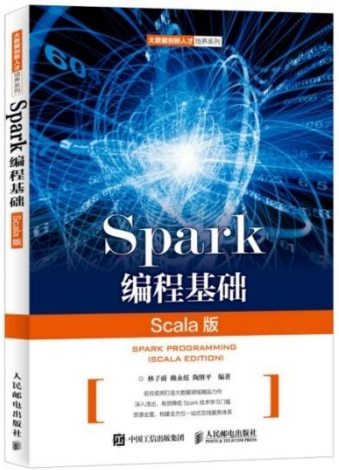
厦门大学林子雨编著

配套实验指导书


- 步步引导，循序渐进，详尽的安装指南为顺利搭建大数据实验环境铺平道路
- 深入浅出，去粗取精，丰富的代码实例帮助快速掌握大数据基础编程方法
- 精心设计，巧妙融合，八套大数据实验题目促进理论与编程知识的消化和吸收
- 结合理论，联系实际，大数据课程综合实验案例精彩呈现大数据分析全流程

林子雨编著《大数据基础编程、实验和案例教程（第2版）》
清华大学出版社 ISBN:978-7-302-55977-1 定价：69元 2020年10月第2版

# 附录H：《Spark编程基础（Scala版）》




## 《Spark编程基础（Scala版）》

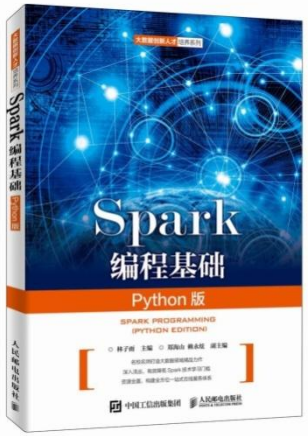
厦门大学 林子雨，赖永炫，陶继平 编著

披荆斩棘，在大数据丛林中开辟学习捷径
填沟削坎，为快速学习Spark技术铺平道路
深入浅出，有效降低Spark技术学习门槛
资源全面，构建全方位一站式在线服务体系

**人民邮电出版社出版发行，ISBN:978-7-115-48816-9**
**教材官网：http://dblab.xmu.edu.cn/post/spark/**

本书以Scala作为开发Spark应用程序的编程语言，系统介绍了Spark编程的基础知识。全书共8章，内容包括大数据技术概述、Scala语言基础、Spark的设计与运行原理、Spark环境搭建和使用方法、RDD编程、Spark SQL、Spark Streaming、Spark MLlib等。本书每个章节都安排了入门级的编程实践操作，以便读者更好地学习和掌握Spark编程方法。本书官网免费提供了全套的在线教学资源，包括讲义PPT、习题、源代码、软件、数据集、授课视频、上机实验指南等。

# 附录I：《Spark编程基础（Python版）》




## 《Spark编程基础（Python版）》

厦门大学 林子雨，郑海山，赖永炫 编著

披荆斩棘，在大数据丛林中开辟学习捷径
填沟削坎，为快速学习Spark技术铺平道路
深入浅出，有效降低Spark技术学习门槛
资源全面，构建全方位一站式在线服务体系

**人民邮电出版社出版发行，ISBN:978-7-115-52439-3**
**教材官网： http://dblab.xmu.edu.cn/post/spark-python/**

本书以Python作为开发Spark应用程序的编程语言，系统介绍了Spark编程的基础知识。全书共8章，内容包括大数据技术概述、Spark的设计与运行原理、Spark环境搭建和使用方法、RDD编程、Spark SQL、Spark Streaming、Structured Streaming、Spark MLlib等。本书每个章节都安排了入门级的编程实践操作，以便读者更好地学习和掌握Spark编程方法。本书官网免费提供了全套的在线教学资源，包括讲义PPT、习题、源代码、软件、数据集、上机实验指南等。

# 附录J：高校大数据课程公共服务平台

高校大数据课程

公 共 服 务 平 台

http://dblab.xmu.edu.cn/post/bigdata-teaching-platform/

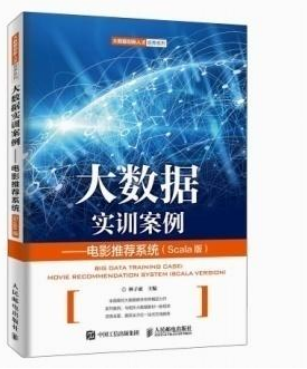
扫一扫访问平台主页


扫一扫观看3分钟FLASH动画宣传片

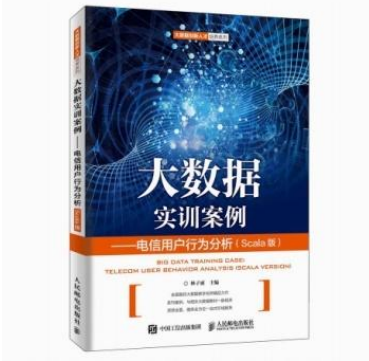
# 附录K：高校大数据实训课程系列案例教材

为了更好满足高校开设大数据实训课程的教材需求，厦门大学数据库实验室林子雨老师团队联合企业共同开发了《高校大数据实训课程系列案例》，目前已经完成开发的系列案例包括：

《电影推荐系统》（已经于2019年5月出版）
《电信用户行为分析》（已经于2019年5月出版）
《实时日志流处理分析》
《微博用户情感分析》
《互联网广告预测分析》
《网站日志处理分析》




系列案例教材将于2019年陆续出版发行，教材相关信息，敬请关注网页后续更新！
http://dblab.xmu.edu.cn/post/shixunkecheng/




扫一扫访问大数据实训课程系列案例教材主页


Thank You!