



实验4：常用HDFS操作

廖 军

重庆大学大数据与软件学院

E-mail: liaojun@cqu.edu.cn

常用的HDFS操作

4.1 HDFS操作常用Shell命令

4.2 利用HDFS的Web管理界面

4.3 HDFS编程实践

具体请参见：

《大数据技术原理与应用第三章 分布式文件系统HDFS 学习指南

访问地址：2版 <http://dblab.xmu.edu.cn/blog/290-2/>

或3版 <http://dblab.xmu.edu.cn/blog/2460-2/#more-2460>

4.1 HDFS操作常用Shell命令

4.1.1 查看命令使用方法

4.1.2 HDFS目录操作

4.1.1 查看命令使用方法

请登录Linux系统，打开一个终端，首先启动Hadoop，命令如下：

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

可以在终端输入如下命令，查看hdfs dfs总共支持哪些操作：

```
$ cd /usr/local/hadoop  
$ ./bin/hdfs dfs
```

可以查看某个命令的作用，比如，当需要查询put命令的具体用法时，可以采用如下命令：

```
$ ./bin/hdfs dfs -help put
```

4.1.2 HDFS 目录操作

1. 目录操作

需要注意的是，Hadoop系统安装好以后，第一次使用HDFS时，需要首先在HDFS中创建用户目录。本教程全部采用hadoop用户登录Linux系统，因此，需要在HDFS中为hadoop用户创建一个用户目录，命令如下：

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

“/user/hadoop”目录就成为hadoop用户对应的用户目录，可以使用如下命令显示HDFS中与当前用户hadoop对应的用户目录下的内容：

```
$ ./bin/hdfs dfs -ls .
```

上面的命令和下面的命令是等价的：

```
$ ./bin/hdfs dfs -ls /user/hadoop
```

4.1.2 HDFS 目录操作

如果要列出HDFS上的所有目录，可以使用如下命令：

```
$ ./bin/hdfs dfs -ls
```

下面，可以使用如下命令创建一个input目录：

```
$ ./bin/hdfs dfs -mkdir input
```

如果要在HDFS的根目录下创建一个名称为input的目录，则需要使用如下命令：

```
$ ./bin/hdfs dfs -mkdir /input
```

可以使用rm命令删除一个目录，比如，可以使用如下命令删除刚才在HDFS中创建的“/input”目录（不是“/user/hadoop/input”目录）：

```
$ ./bin/hdfs dfs -rm -r /input
```

4.1.2 HDFS 目录操作

2. 文件操作

首先，使用vim编辑器，在本地Linux文件系统的“/home/hadoop/”目录下创建一个文件myLocalFile.txt，里面可以随意输入一些单词，比如，输入如下三行：

```
Hadoop  
Spark  
XMU DBLAB
```

然后，可以使用如下命令把本地文件系统的“/home/hadoop/myLocalFile.txt”上传到HDFS中的当前用户目录的input目录下，也就是上传到HDFS的“/user/hadoop/input/”目录下：

```
$ ./bin/hdfs dfs -put /home/hadoop/myLocalFile.txt input
```

可以使用ls命令查看一下文件是否成功上传到HDFS中，具体如下：

```
$ ./bin/hdfs dfs -ls input
```

4.1.2 HDFS 目录操作

下面使用如下命令查看HDFS中的myLocalFile.txt这个文件的内容：

```
$ ./bin/hdfs dfs -cat input/myLocalFile.txt
```

下面把HDFS中的myLocalFile.txt文件下载到本地文件系统中的“/home/hadoop/下载”这个目录下，命令如下：

```
$ ./bin/hdfs dfs -get input/myLocalFile.txt /home/hadoop/下载
```

可以使用如下命令，到本地文件系统查看下载下来的文件myLocalFile.txt：

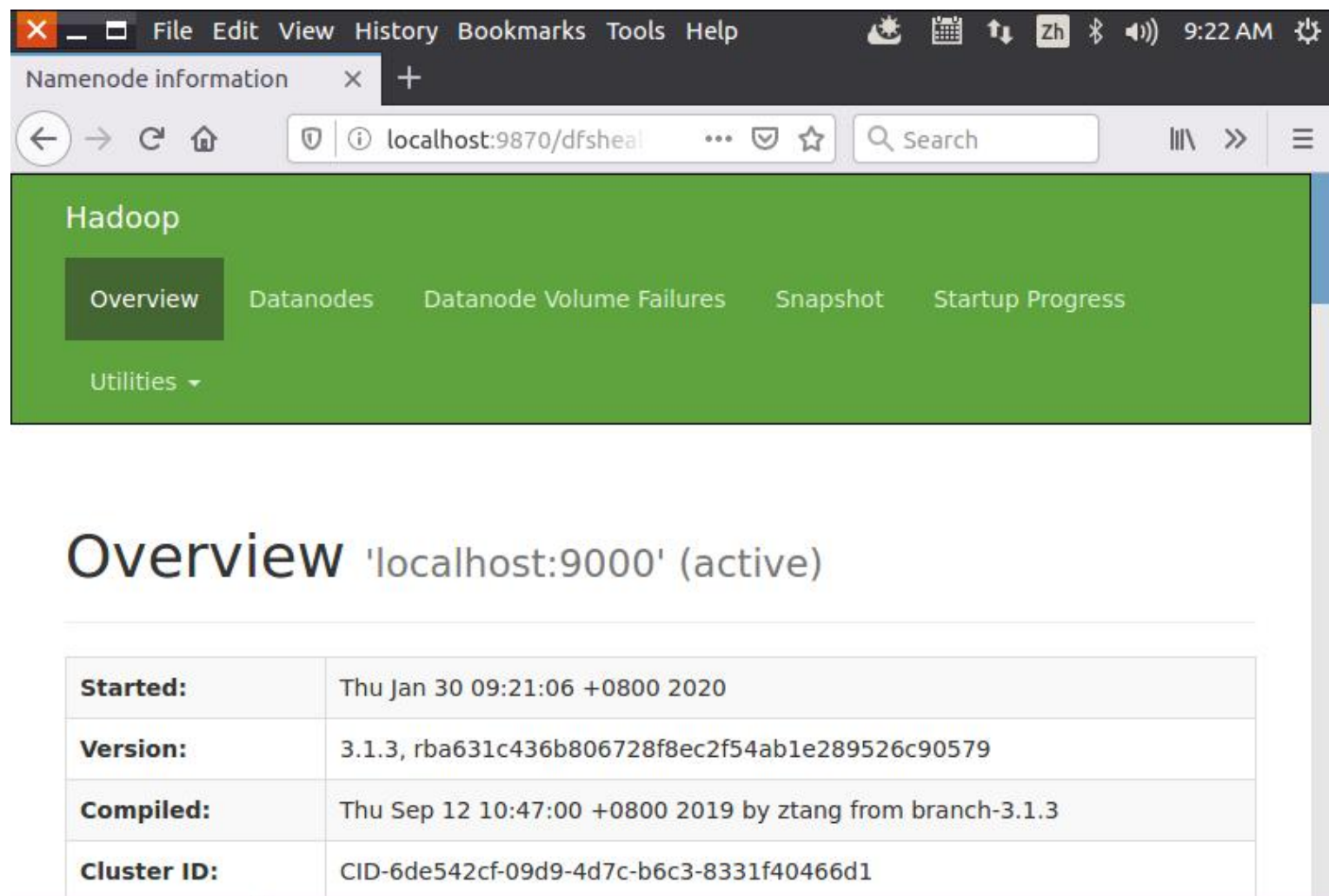
```
$ cd ~  
$ cd 下载  
$ ls  
$ cat myLocalFile.txt
```

4.1.2 HDFS 目录操作

最后，了解一下如何把文件从HDFS中的一个目录拷贝到HDFS中的另外一个目录。比如，如果要把HDFS的“/user/hadoop/input/myLocalFile.txt”文件，拷贝到HDFS的另外一个目录“/input”中（注意，这个input目录位于HDFS根目录下），可以使用如下命令：

```
$ ./bin/hdfs dfs -cp input/myLocalFile.txt /input
```


4.2 利用HDFS的Web管理界面



The screenshot displays a web browser window with the Hadoop Web Management Interface. The browser's address bar shows the URL `localhost:9870/dfshealth.html`. The interface has a green header bar with the word "Hadoop" and a navigation menu with tabs: "Overview" (selected), "Datanodes", "Datanode Volume Failures", "Snapshot", and "Startup Progress". Below the tabs is a "Utilities" dropdown menu. The main content area is titled "Overview 'localhost:9000' (active)" and contains a table with the following information:

Started:	Thu Jan 30 09:21:06 +0800 2020
Version:	3.1.3, rba631c436b806728f8ec2f54ab1e289526c90579
Compiled:	Thu Sep 12 10:47:00 +0800 2019 by ztang from branch-3.1.3
Cluster ID:	CID-6de542cf-09d9-4d7c-b6c3-8331f40466d1

4.3 HDFS编程实践

现在要执行的任务是：假设在目录 “hdfs://localhost:9000/user/hadoop” 下面有几个文件，分别是file1.txt、file2.txt、file3.txt、file4.abc和file5.abc，这里需要从该目录中过滤出所有后缀名不为 “.abc” 的文件，对过滤之后的文件进行读取，并将这些文件的内容合并到文件

“hdfs://localhost:9000/user/hadoop/merge.txt” 中。

4.3.1 在Eclipse中创建项目

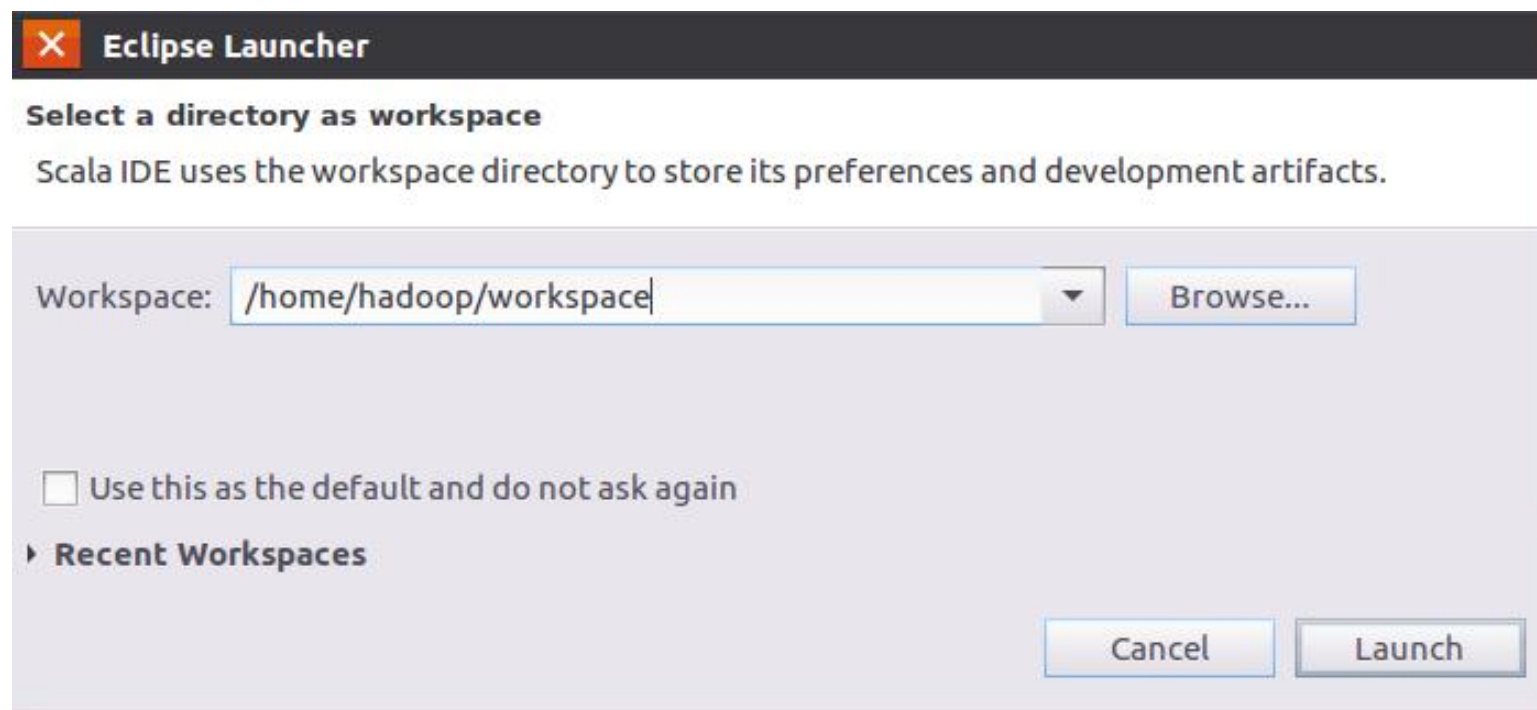
4.3.2为项目添加需要用到的JAR包

4.3.3 编写Java应用程序

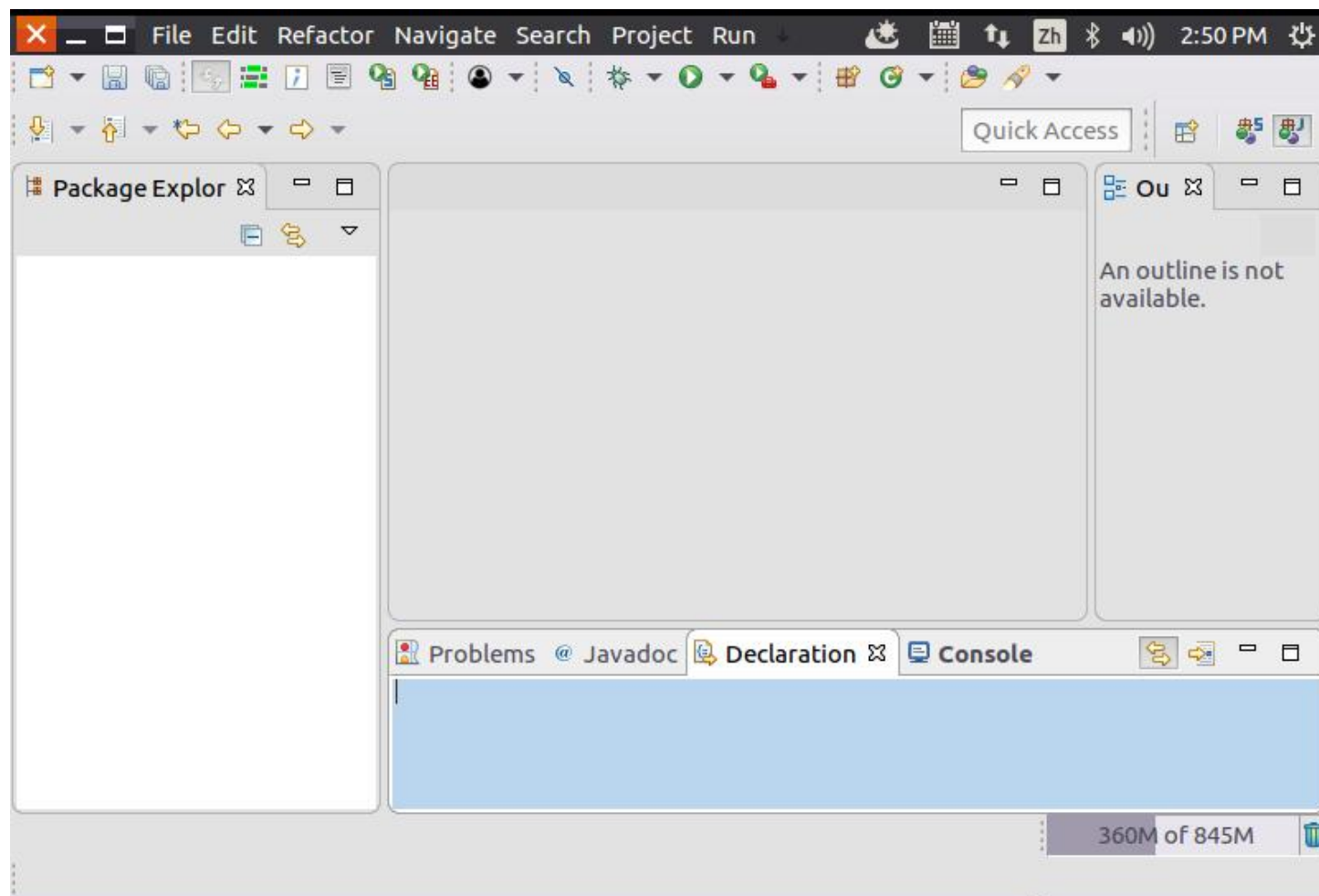
4.3.4 编译运行程序

4.3.5应用程序的部署

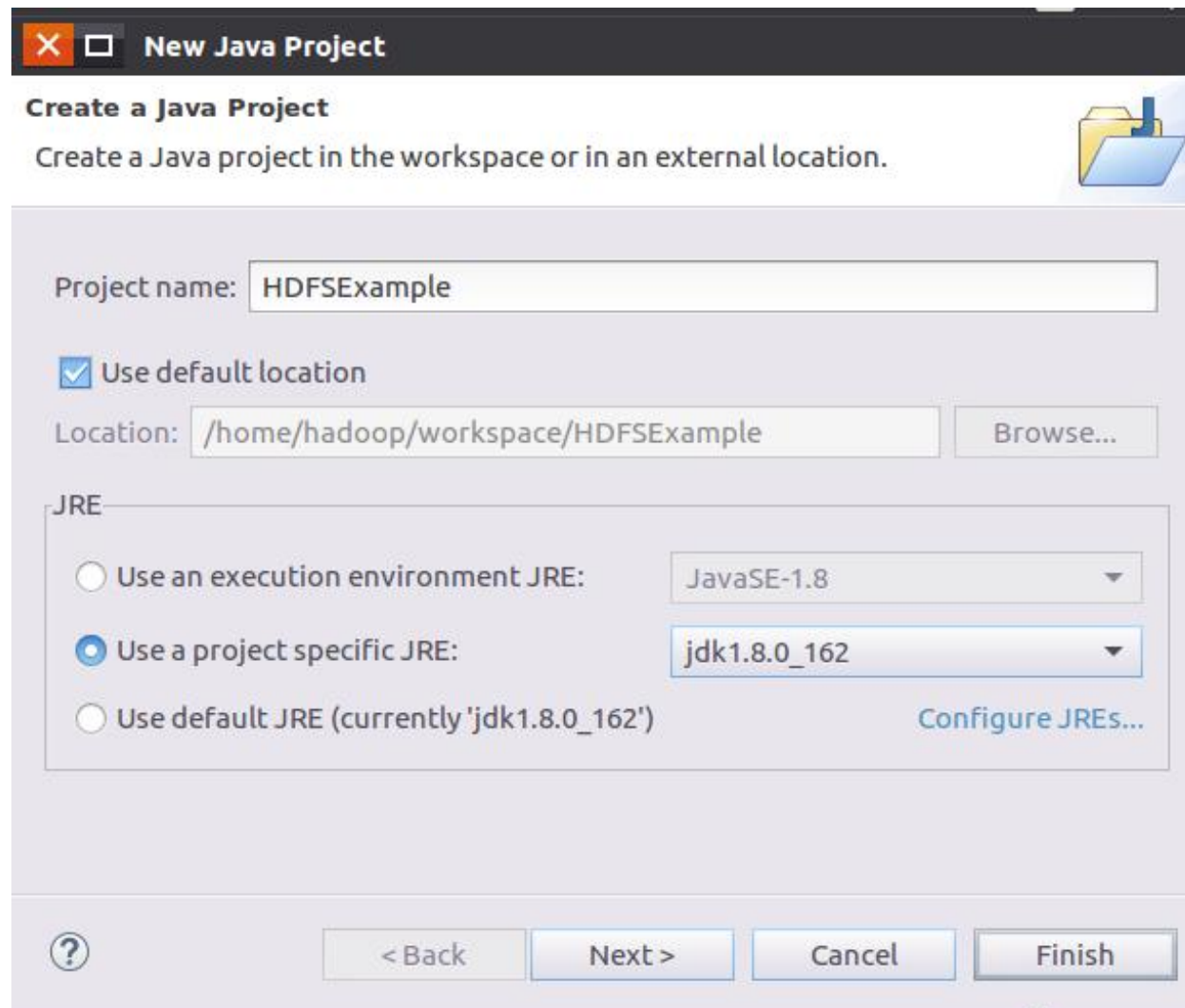
4.3.1 在Eclipse中创建项目



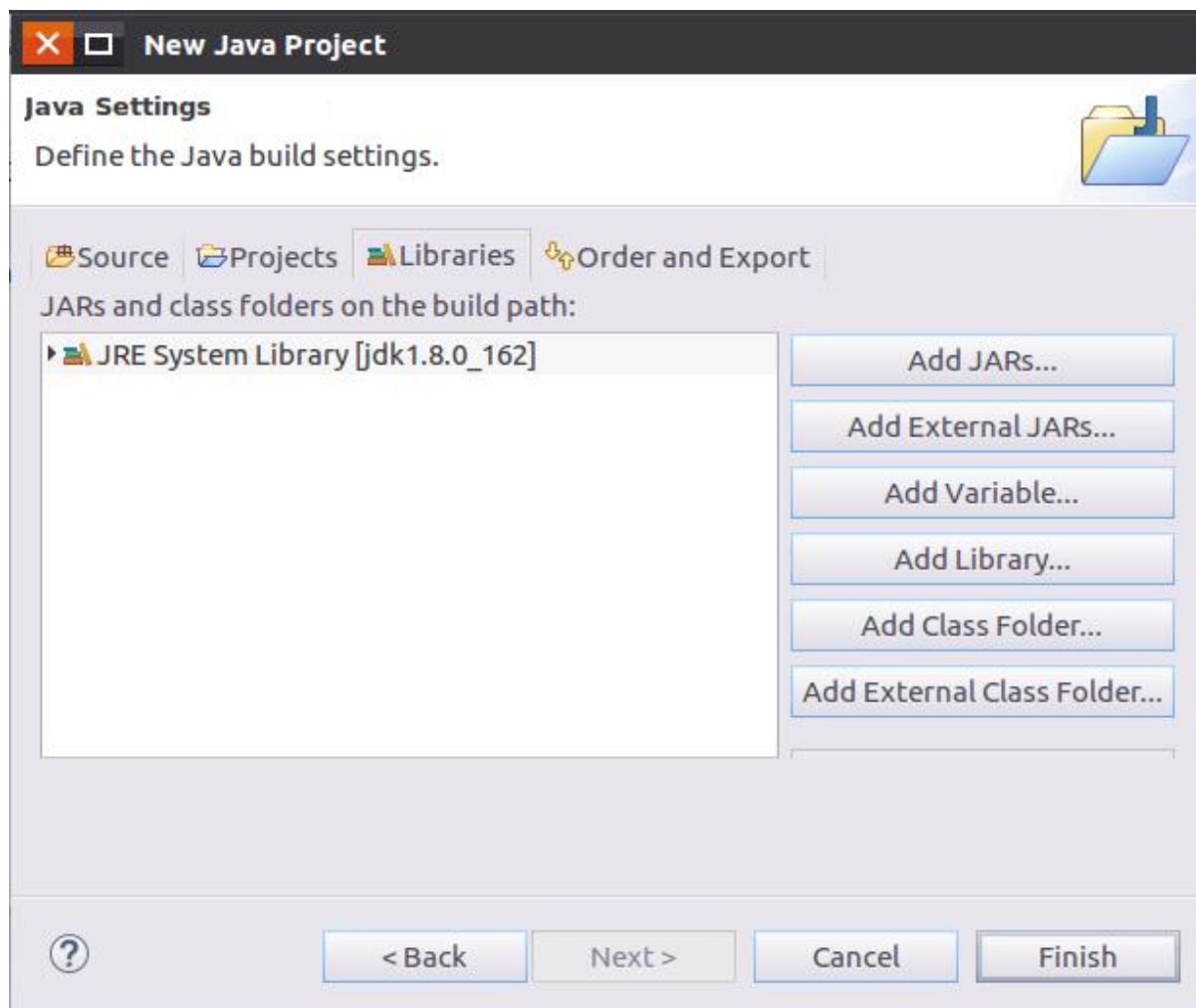
4.3.1 在Eclipse中创建项目



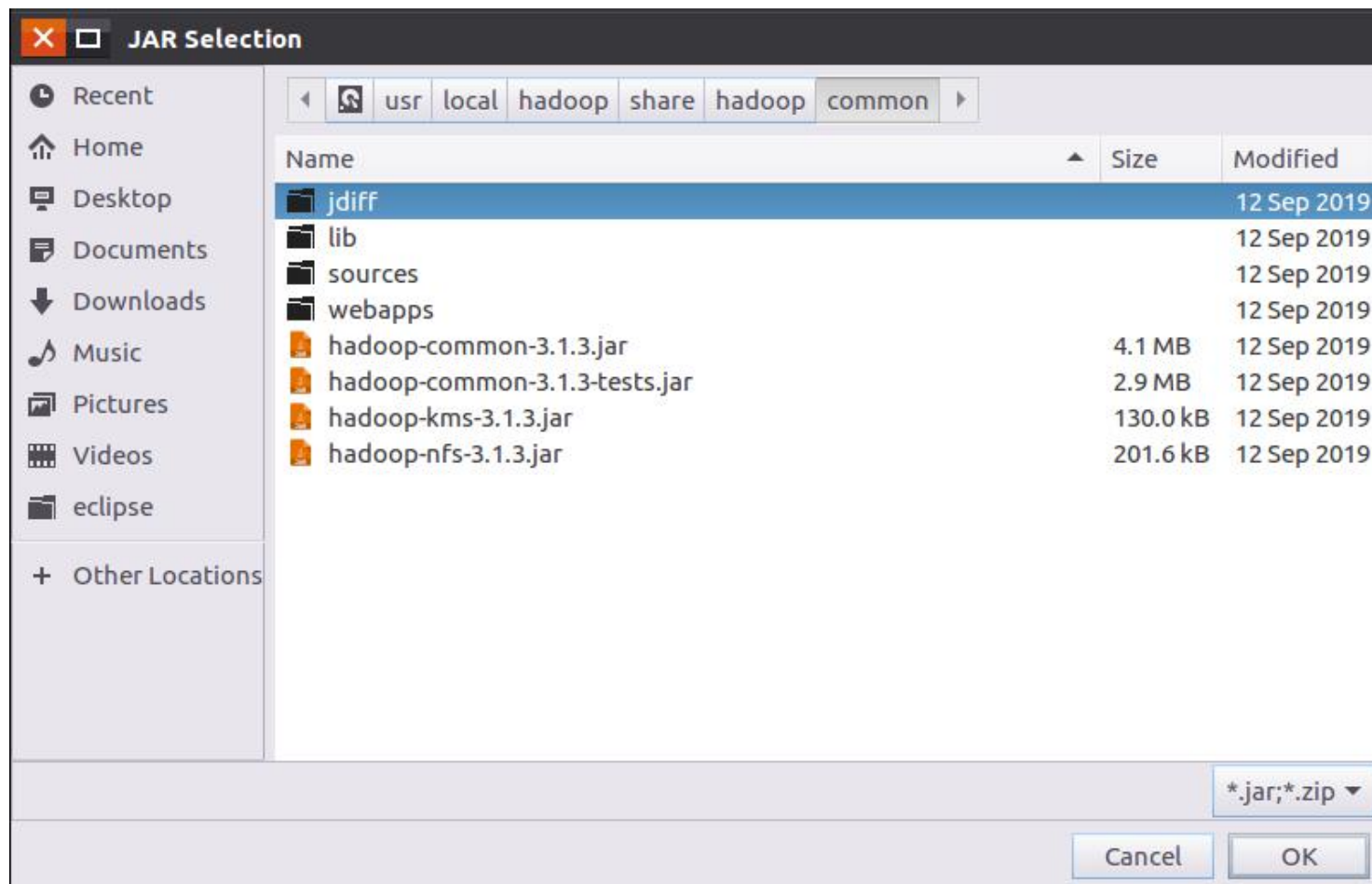
4.3.1 在Eclipse中创建项目



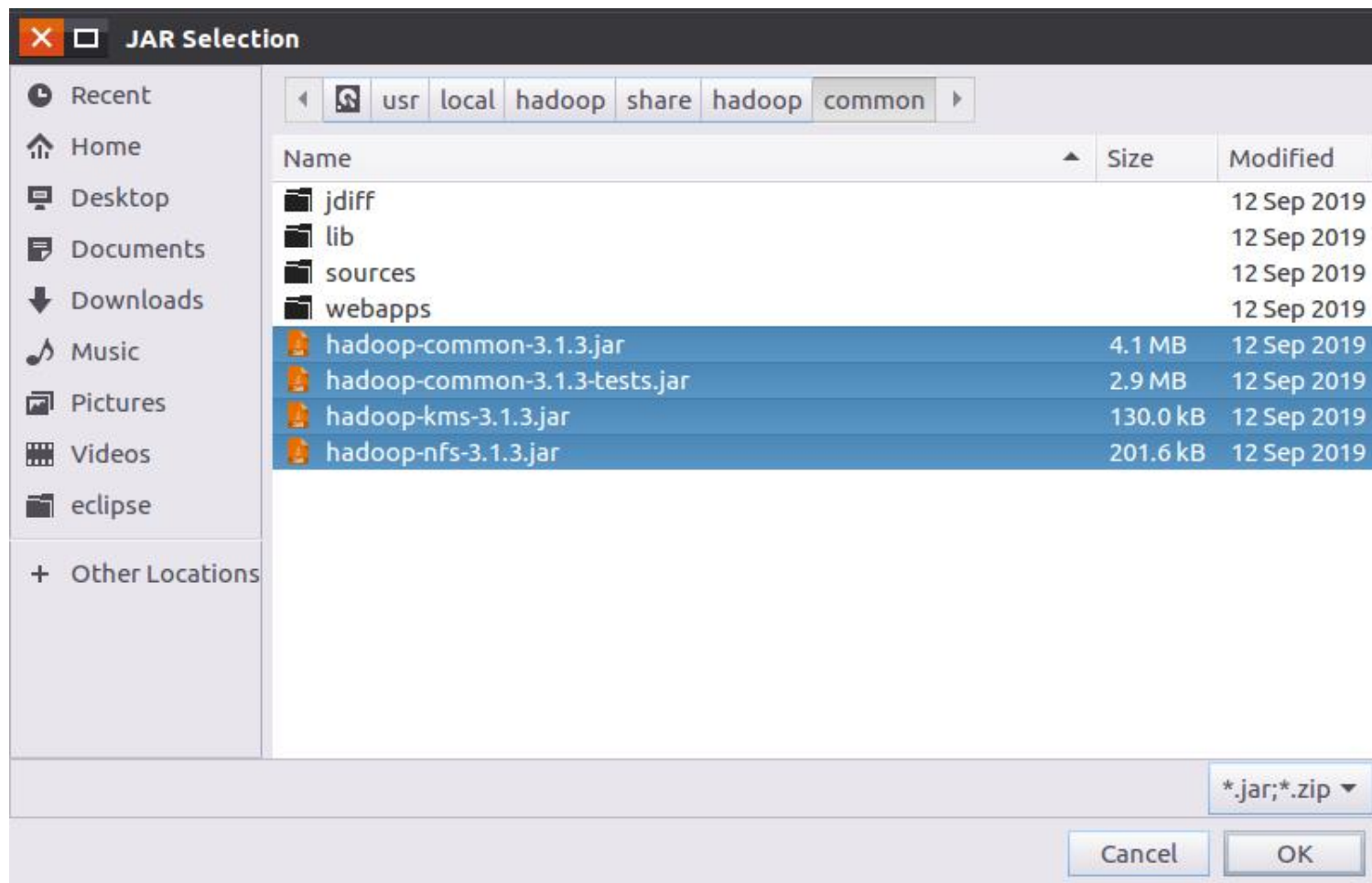
4.3.2 为项目添加需要用到的JAR包



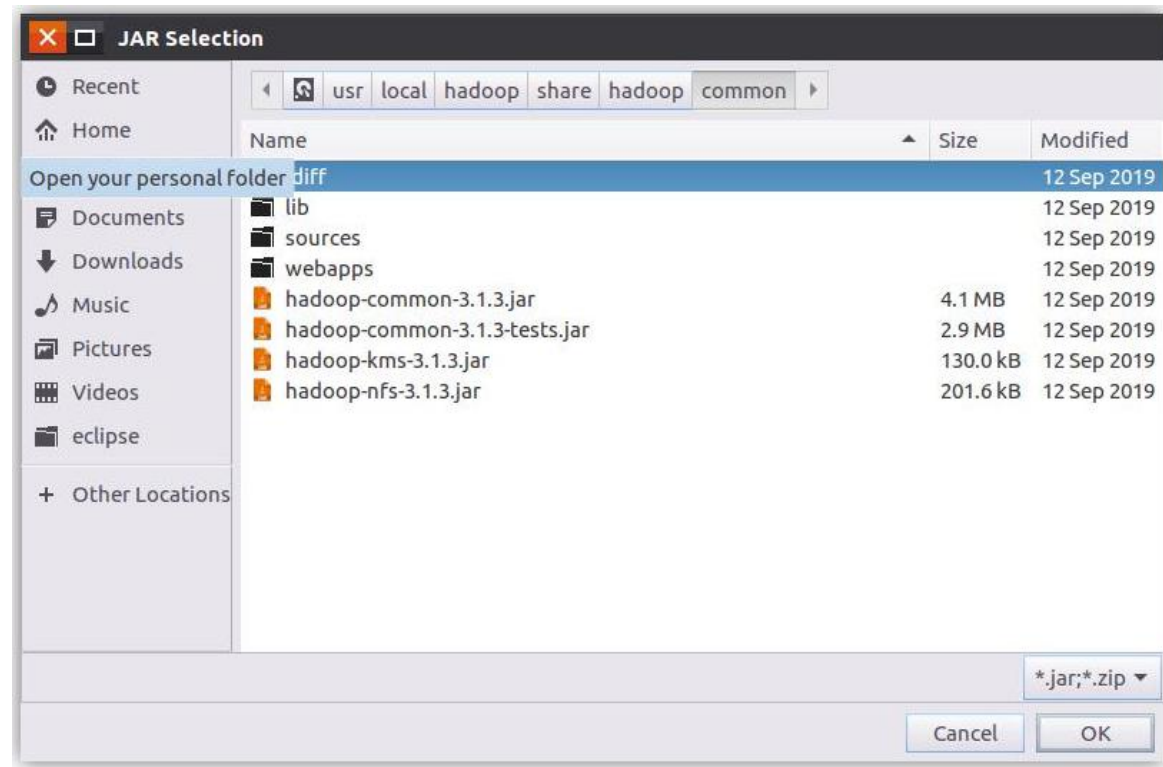
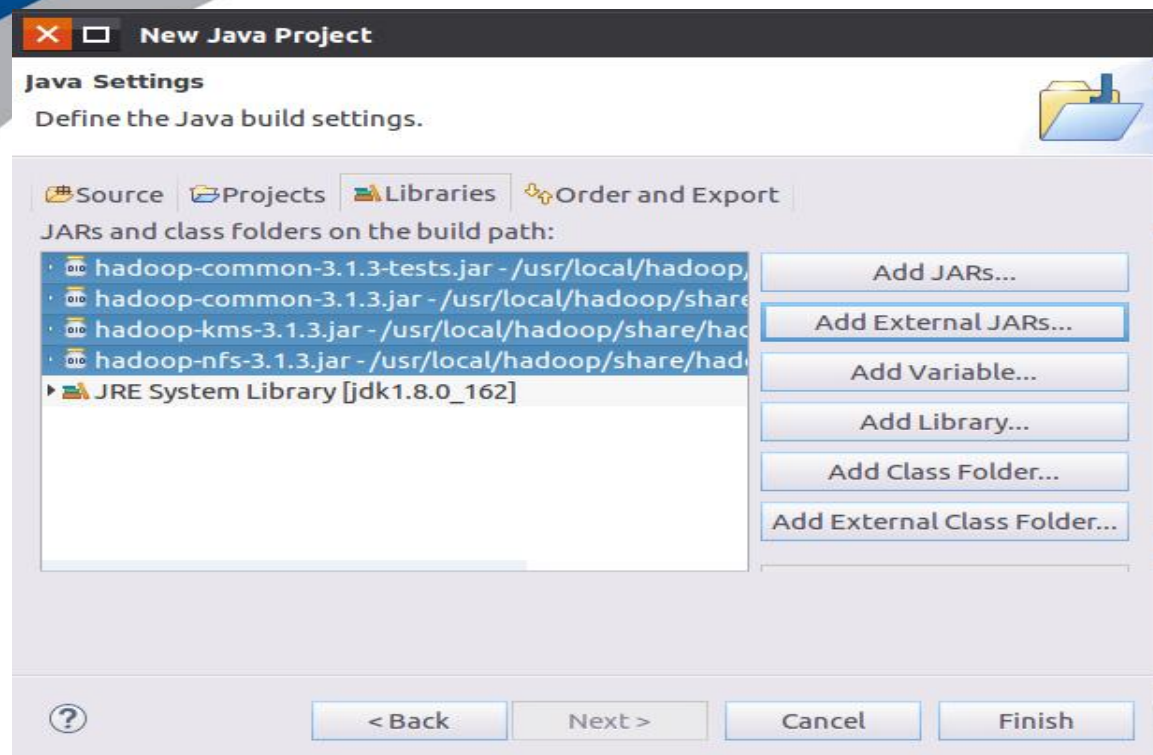
4.3.2 为项目添加需要用到的JAR包



4.3.2 为项目添加需要用到的JAR包

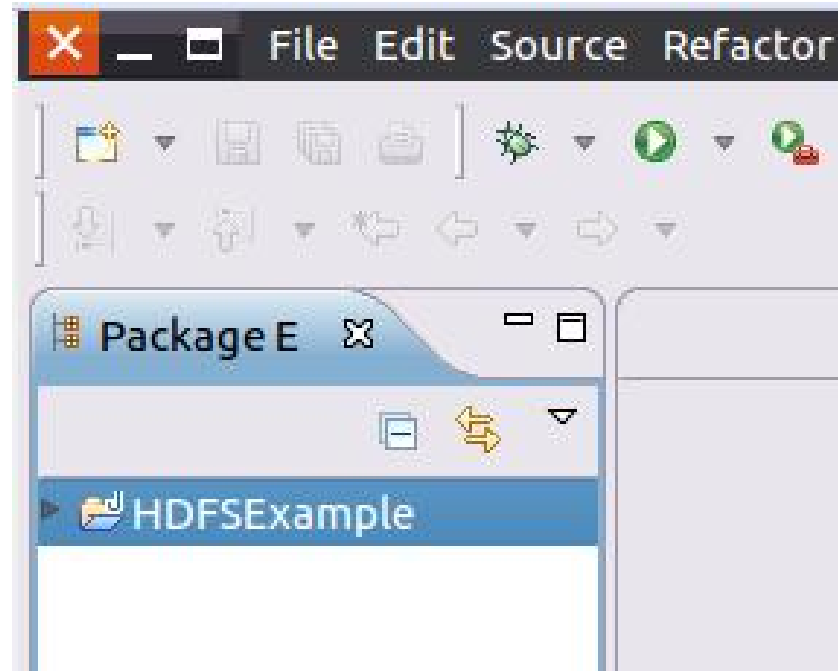


4.3.2为项目添加需要用到的JAR包



- 1) “/usr/local/hadoop/share/hadoop/common” 目录下的所有JAR包，包括hadoop-common-3.1.3.jar、hadoop-common-3.1.3-tests.jar、hadoop-nfs-3.1.3.jar和hadoop-kms-3.1.3.jar，
- (2) “/usr/local/hadoop/share/hadoop/common/lib” 目录下的所有JAR包
- (3) “/usr/local/hadoop/share/hadoop/hdfs” 目录下的所有JAR包
- (4) “/usr/local/hadoop/share/hadoop/hdfs/lib” 目录下的所有JAR包。

4.3.3 编写Java应用程序



4.3.3 编写Java应用程序

New Java Class

Java Class

⚠ The use of the default package is discouraged.

Source folder: HDFSEExample/src Browse...

Package: (default) Browse...

☐ Enclosing type: Browse...

Name: MergeFile

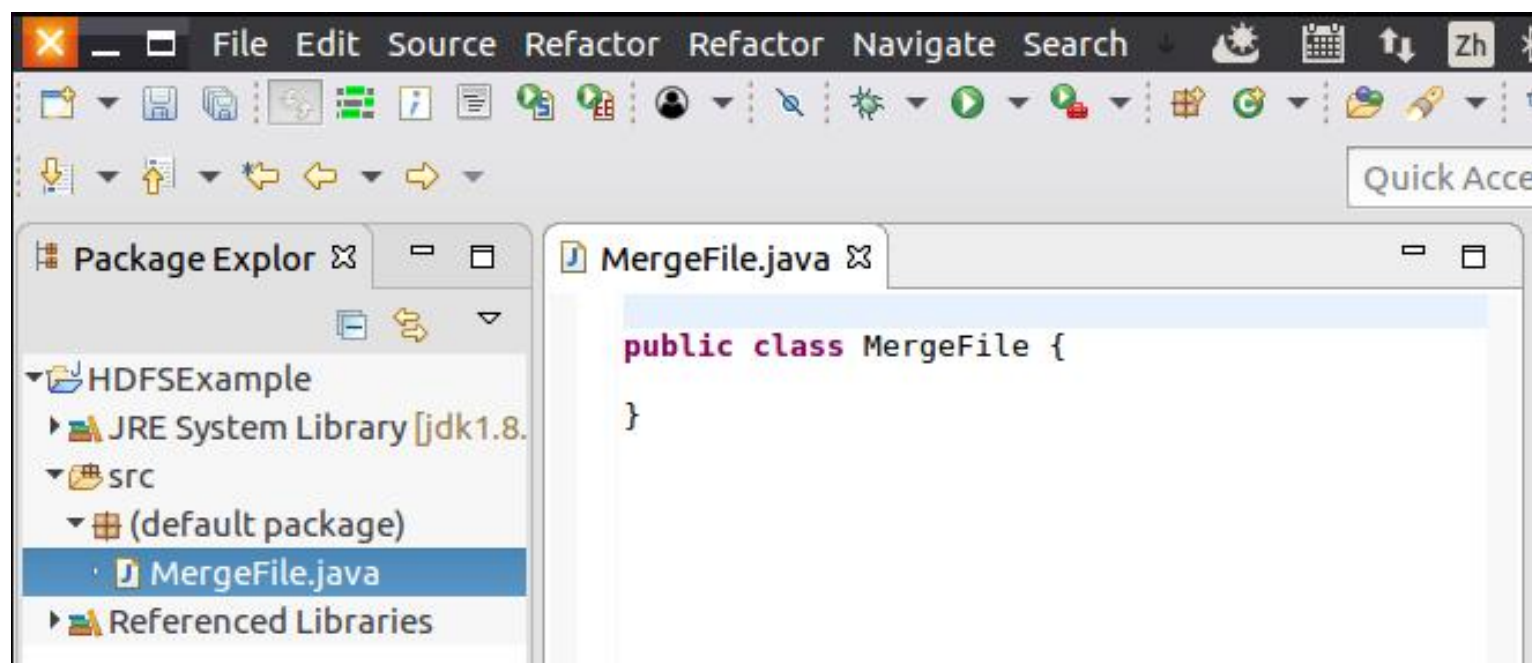
Modifiers: ☒ public ☐ package ☐ private ☐ protected
☐ abstract ☐ final ☐ static

Superclass: java.lang.Object Browse...

Interfaces: Add...

? Cancel Finish

4.3.3 编写Java应用程序



4.3.3 编写Java应用程序

```
import java.io.IOException;
import java.io.PrintStream;
import java.net.URI;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.*;

/**
 * 过滤掉文件名满足特定条件的文件
 */
class MyPathFilter implements PathFilter {
    String reg = null;
    MyPathFilter(String reg) {
        this.reg = reg;
    }
    public boolean accept(Path path) {
        if (!(path.toString().matches(reg)))
            return true;
        return false;
    }
}
```

4.3.3 编写Java应用程序

```
/**
 * 利用FSDDataOutputStream和FSDDataInputStream合并HDFS中的文件
 */
public class MergeFile {
    Path inputPath = null; //待合并的文件所在的目录的路径
    Path outputPath = null; //输出文件的路径
    public MergeFile(String input, String output) {
        this.inputPath = new Path(input);
        this.outputPath = new Path(output);
    }
    public void doMerge() throws IOException {
        Configuration conf = new Configuration();
        conf.set("fs.defaultFS", "hdfs://localhost:9000");
        conf.set("fs.hdfs.impl", "org.apache.hadoop.hdfs.DistributedFileSystem");
        FileSystem fsSource =
        FileSystem.get(URI.create(inputPath.toString()), conf);
```

4.3.3 编写Java应用程序

```
FileSystem fsDst = FileSystem.get(URI.create(outputPath.toString()), conf);
//下面过滤掉输入目录中后缀为.abc的文件
FileStatus[] sourceStatus = fsSource.listStatus(inputPath,
    new MyPathFilter(".*\\.abc"));
FSDataOutputStream fsdos = fsDst.create(outputPath);
PrintStream ps = new PrintStream(System.out);
//下面分别读取过滤之后的每个文件的内容，并输出到同一个文件中
for (FileStatus sta : sourceStatus) {
    //下面打印后缀不为.abc的文件的路径、文件大小
    System.out.print("路径: " + sta.getPath() + " 文件大小: " +
        sta.getLen()
        + " 权限: " + sta.getPermission() + " 内容:");

    FSDataInputStream fsdis = fsSource.open(sta.getPath());
    byte[] data = new byte[1024];
    int read = -1;

    while ((read = fsdis.read(data)) > 0) {
        ps.write(data, 0, read);
        fsdos.write(data, 0, read);
    }
    fsdis.close();
}
ps.close();
fsdos.close();
```


4.3.3 编写Java应用程序

```
public static void main(String[] args) throws IOException {  
    MergeFile merge = new MergeFile(  
        "hdfs://localhost:9000/user/hadoop/",  
        "hdfs://localhost:9000/user/hadoop/merge.txt");  
    merge.doMerge();  
}  
}
```


4.3.4 编译运行程序

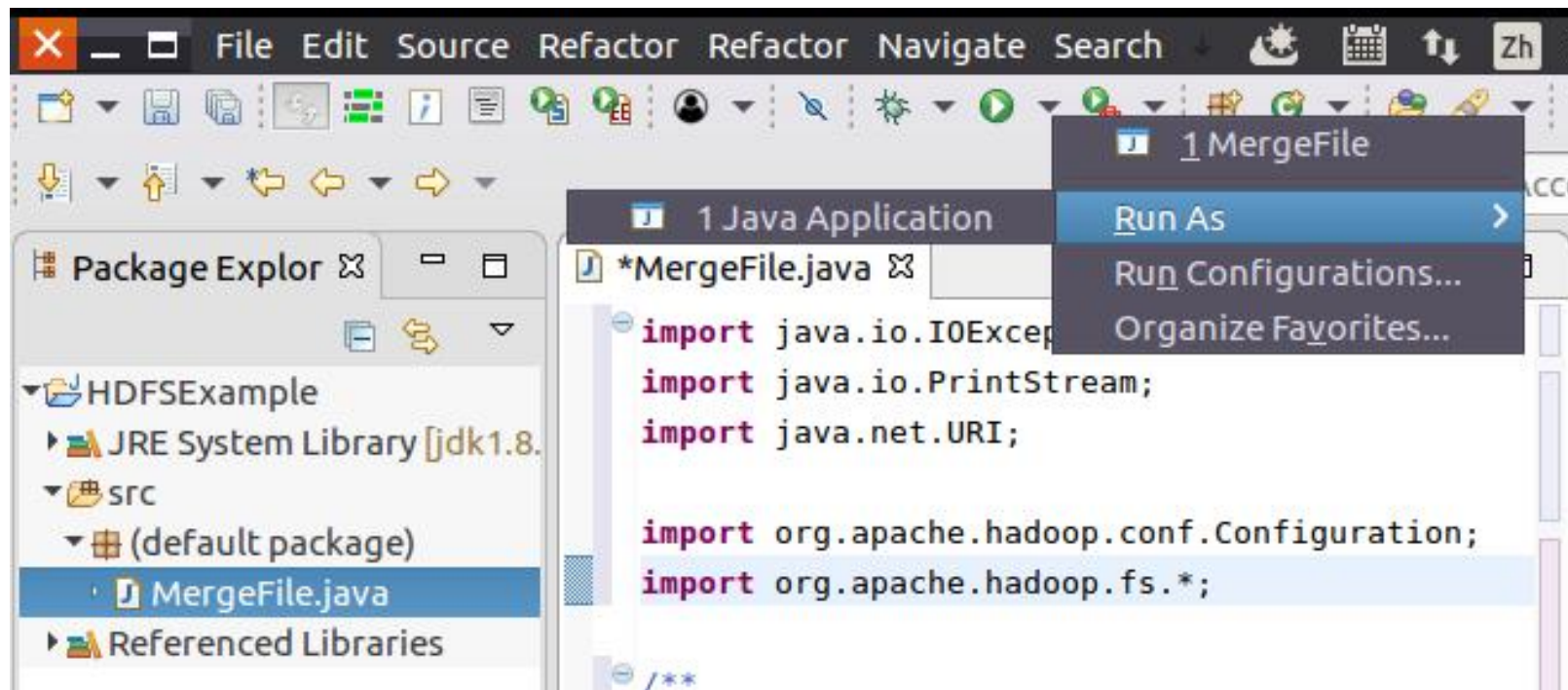
在开始编译运行程序之前，请一定确保Hadoop已经启动运行，如果还没有启动，需要打开一个Linux终端，输入以下命令启动Hadoop：

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

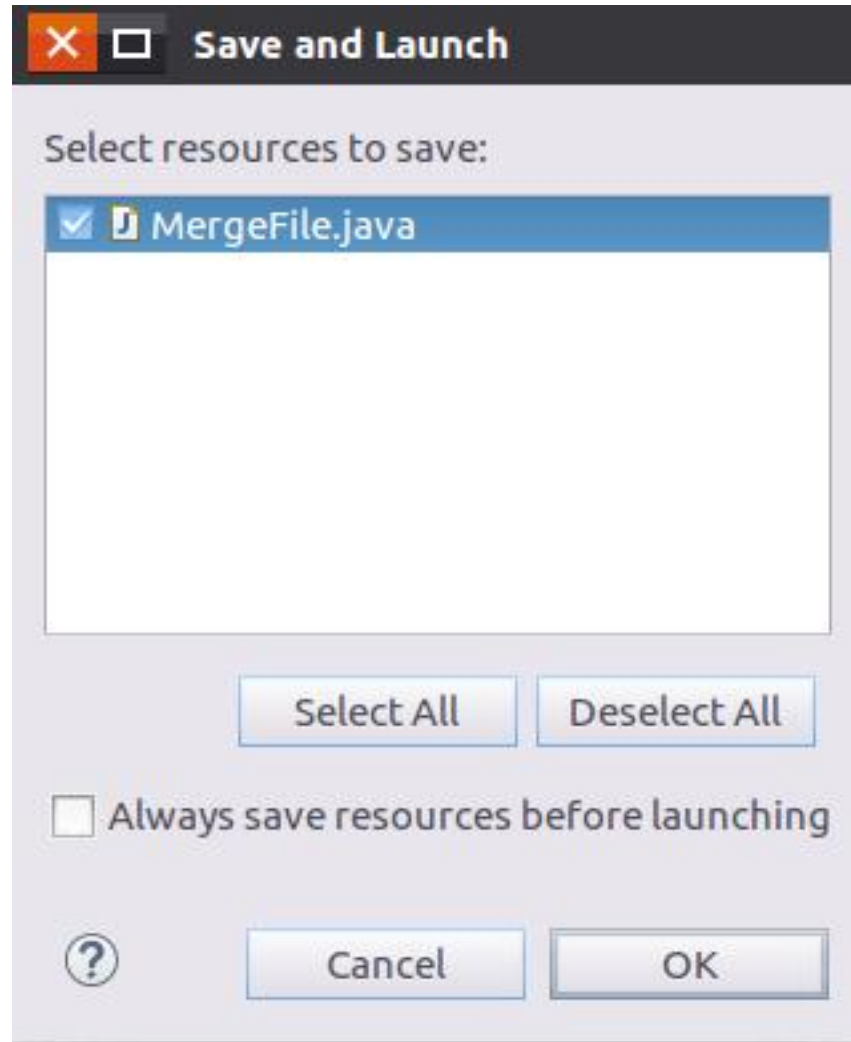
表4-1 HDFS系统中的文件内容

文件名称	文件内容
file1.txt	this is file1.txt
file2.txt	this is file2.txt
file3.txt	this is file3.txt
file4.abc	this is file4.abc
file5.abc	this is file5.abc

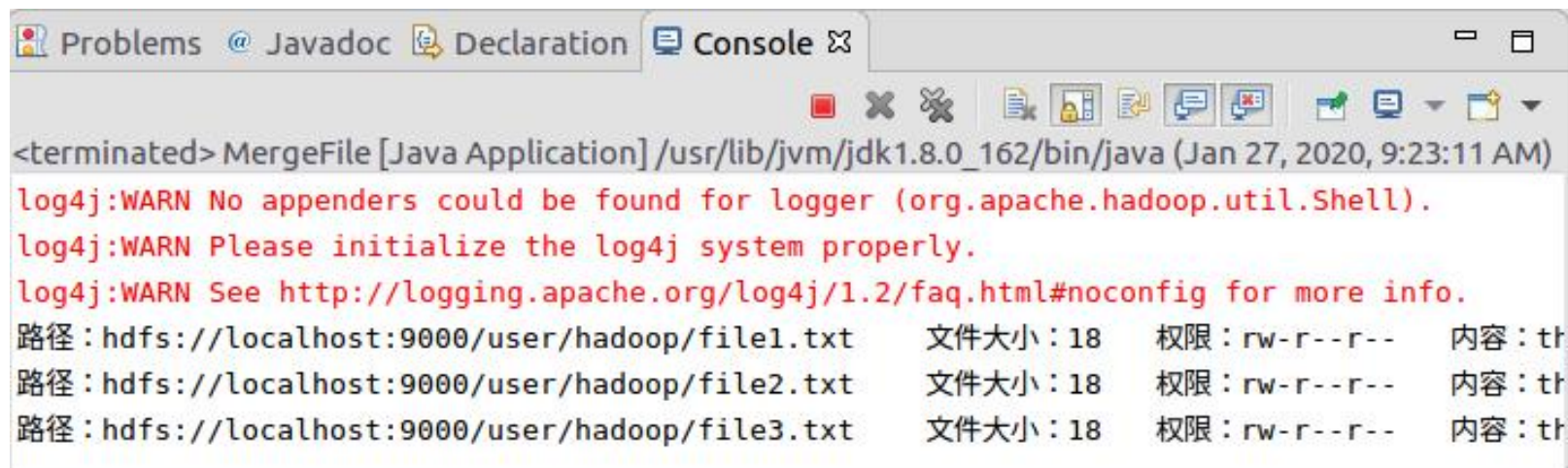
4.3.4 编译运行程序



4.3.4 编译运行程序



4.3.4 编译运行程序



The screenshot shows an IDE console window with the following content:

```
<terminated> MergeFile [Java Application] /usr/lib/jvm/jdk1.8.0_162/bin/java (Jan 27, 2020, 9:23:11 AM)
log4j:WARN No appenders could be found for logger (org.apache.hadoop.util.Shell).
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.
路径：hdfs://localhost:9000/user/hadoop/file1.txt      文件大小：18      权限：rw-r--r--      内容：th
路径：hdfs://localhost:9000/user/hadoop/file2.txt      文件大小：18      权限：rw-r--r--      内容：th
路径：hdfs://localhost:9000/user/hadoop/file3.txt      文件大小：18      权限：rw-r--r--      内容：th
```

4.3.4 编译运行程序

如果程序运行成功，这时，可以到HDFS中查看生成的merge.txt文件，比如，可以在Linux终端中执行如下命令：

```
$ cd /usr/local/hadoop  
$ ./bin/hdfs dfs -ls /user/hadoop  
$ ./bin/hdfs dfs -cat /user/hadoop/merge.txt
```

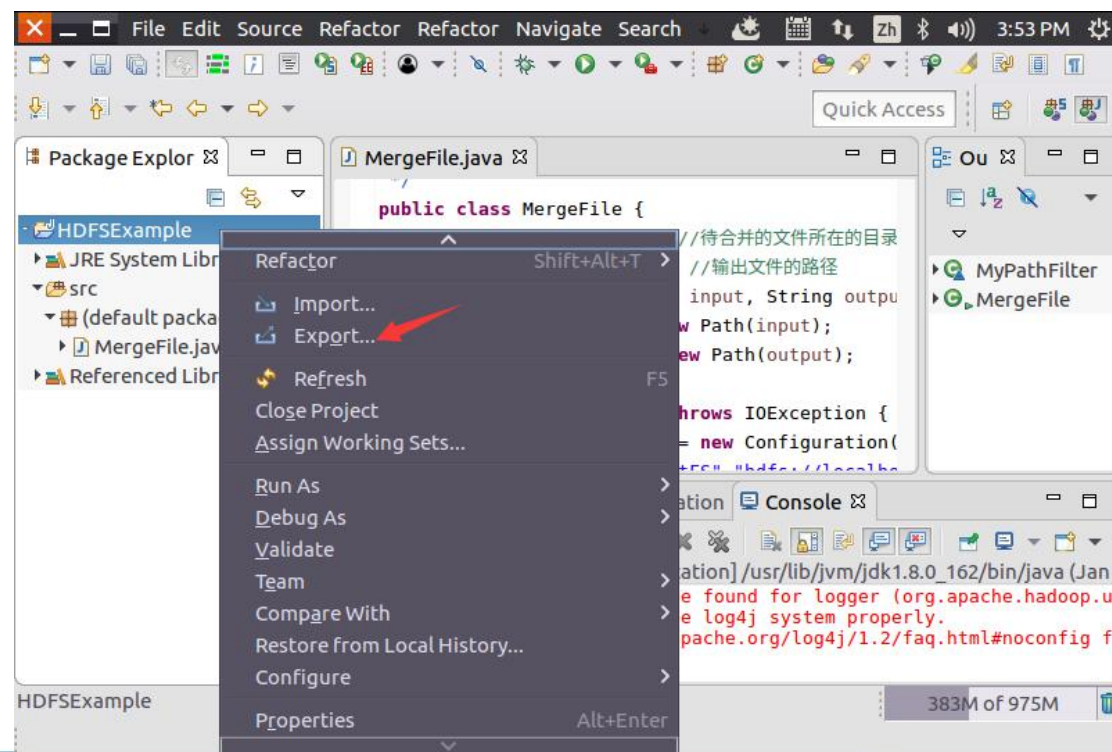
可以看到如下结果：

```
this is file1.txt  
this is file2.txt  
this is file3.txt
```

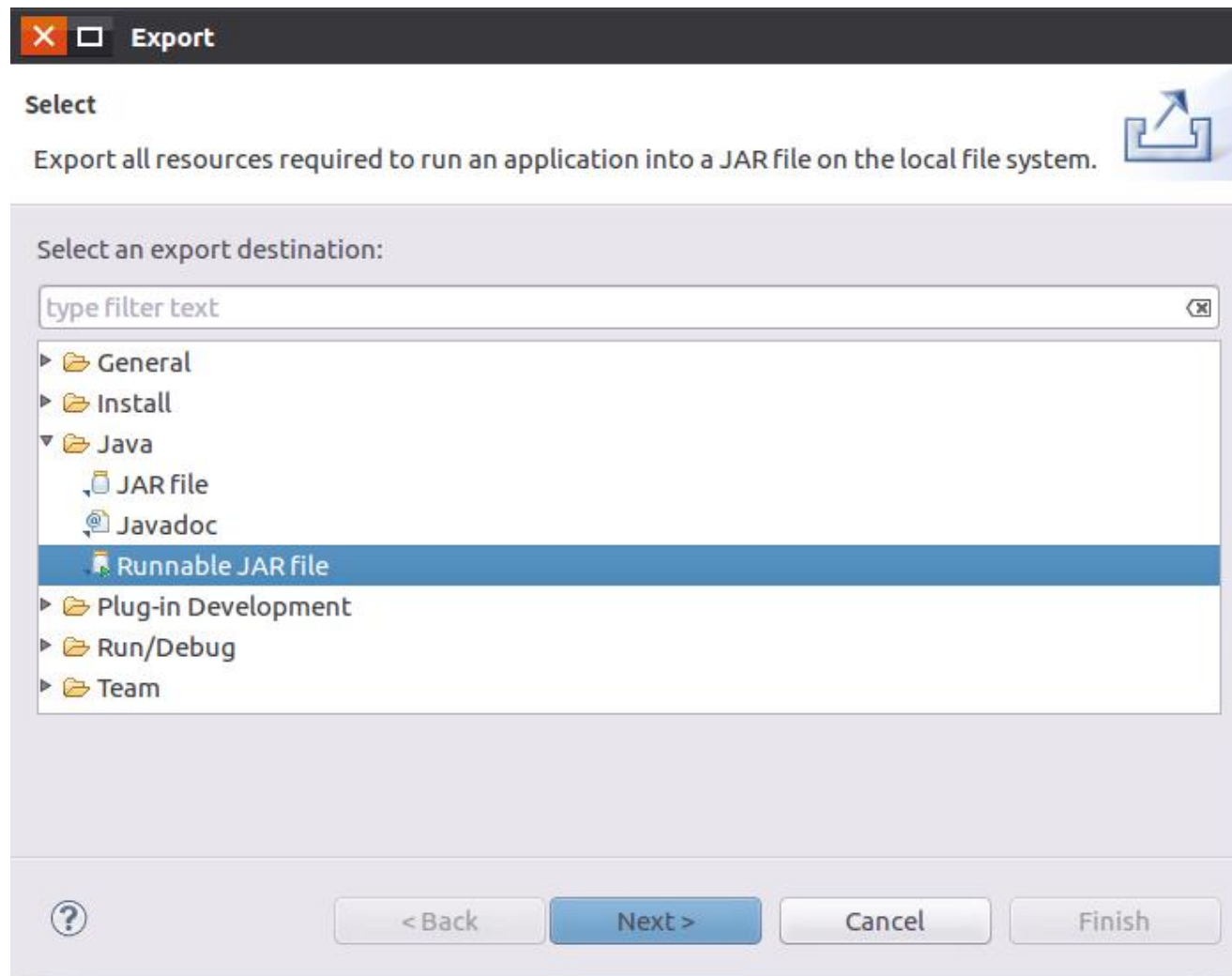
4.3.5 应用程序的部署

首先，在Hadoop安装目录下新建一个名称为myapp的目录，用来存放我们自己编写的Hadoop应用程序，可以在Linux的终端中执行如下命令：

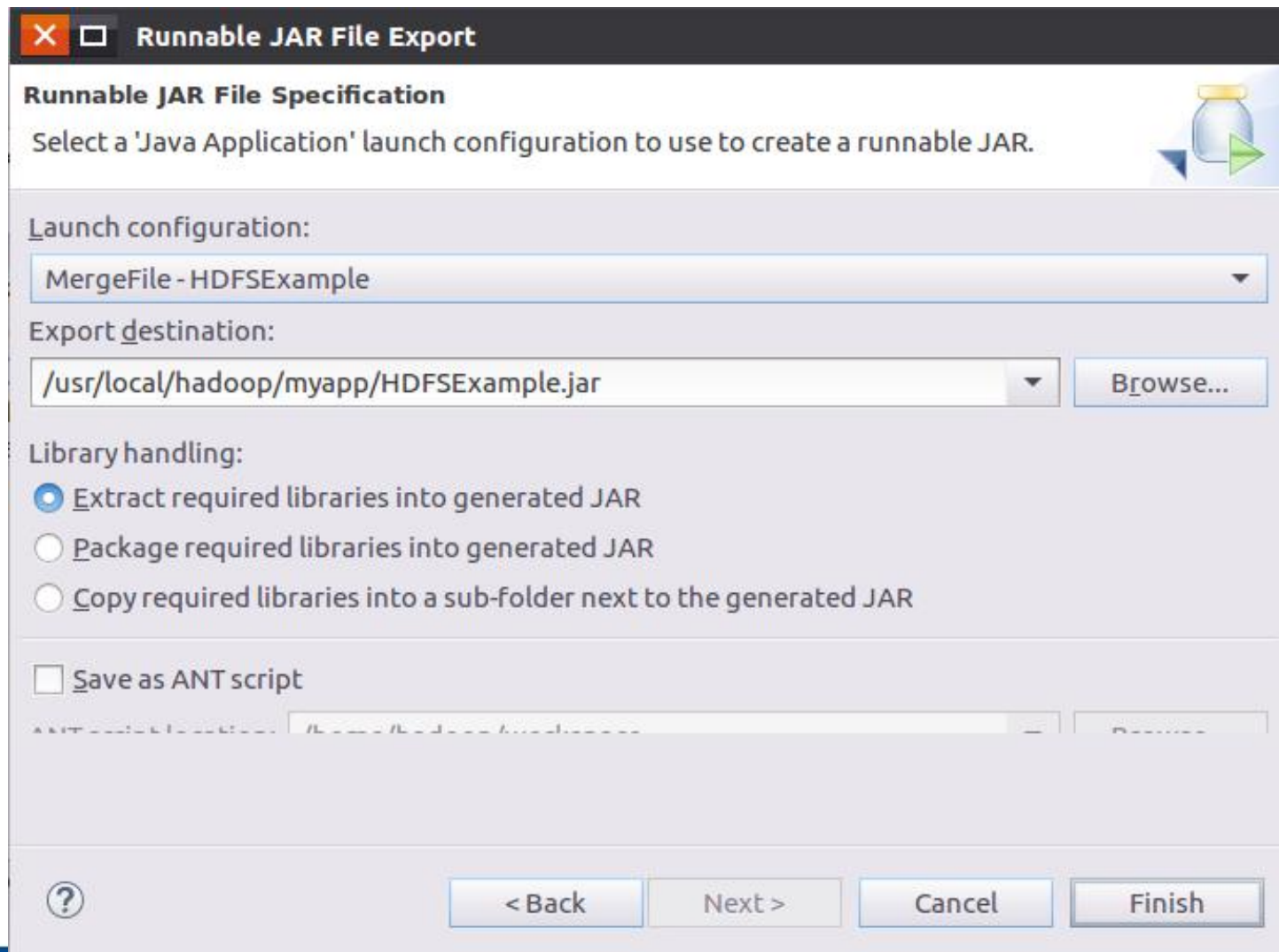
```
$ cd /usr/local/hadoop  
$ mkdir myapp
```



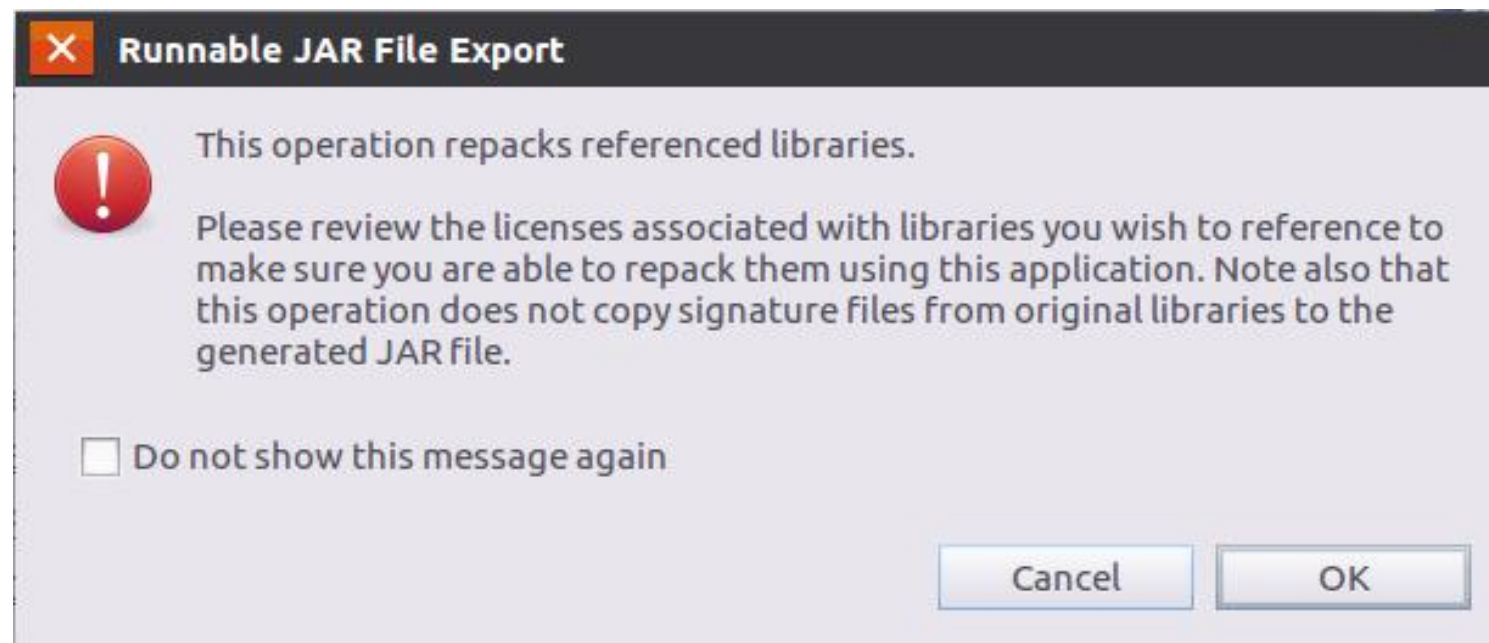
4.3.5 应用程序的部署



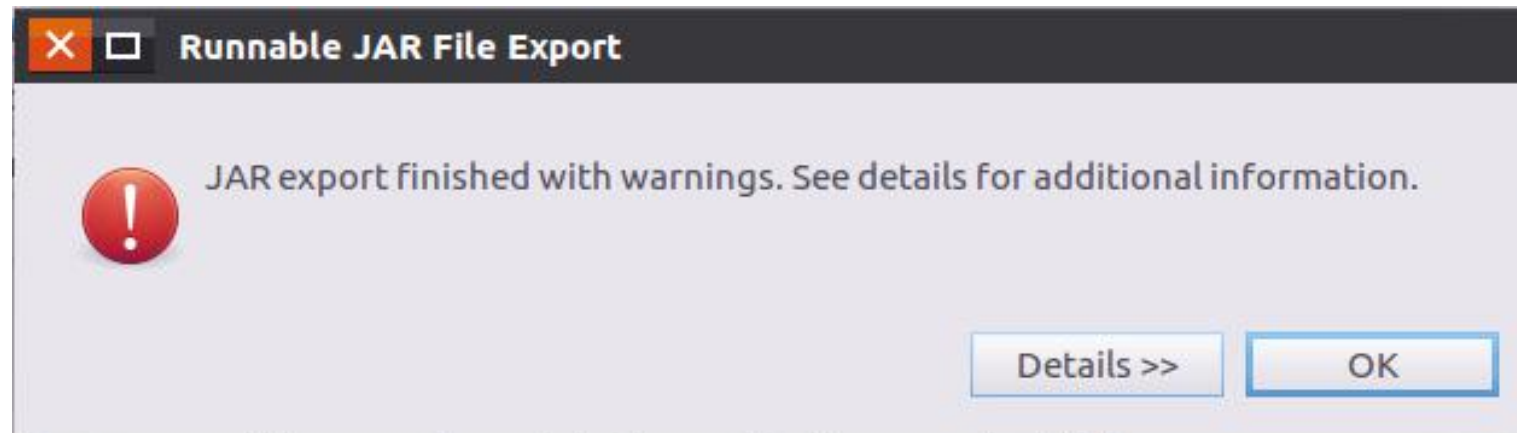
4.3.5 应用程序的部署



4.3.5 应用程序的部署



4.3.5 应用程序的部署



4.3.5 应用程序的部署

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

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

可以看到，“/usr/local/hadoop/myapp”目录下已经存在一个HDFSExample.jar文件。由于之前已经运行过一次程序，已经生成了merge.txt，因此，需要首先执行如下命令删除该文件：

```
$ cd /usr/local/hadoop  
$ ./bin/hdfs dfs -rm /user/hadoop/merge.txt
```

4.3.5 应用程序的部署

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/HDFSExample.jar
```

上面程序执行结束以后，可以到HDFS中查看生成的merge.txt文件，比如，可以在Linux终端中执行如下命令：

```
$ cd /usr/local/hadoop  
$ ./bin/hdfs dfs -ls /user/hadoop  
$ ./bin/hdfs dfs -cat /user/hadoop/merge.txt
```

可以看到如下结果：

```
this is file1.txt  
this is file2.txt  
this is file3.txt
```

4.4 本章小结

大数据时代必须解决海量数据的高效存储问题，为此，谷歌开发了分布式文件系统GFS，通过网络实现文件在多台机器上的分布式存储，较好地满足了大规模数据存储的需求。HDFS是针对GFS的开源实现，它是Hadoop两大核心组成部分之一。

在很多情形下，需要使用Shell命令来操作HDFS，因此，本章介绍了HDFS操作常用的Shell命令，包括目录操作命令和文件操作命令等。同时，还介绍了如何利用HDFS的Web管理界面，以可视化的方式查看HDFS的相关信息。最后，详细介绍了如何使用Eclipse开发操作HDFS的Java应用程序。介绍的Eclipse开发方法，为后续章节的编程开发提供了很好的借鉴。