CS590RA
Tutorial 2

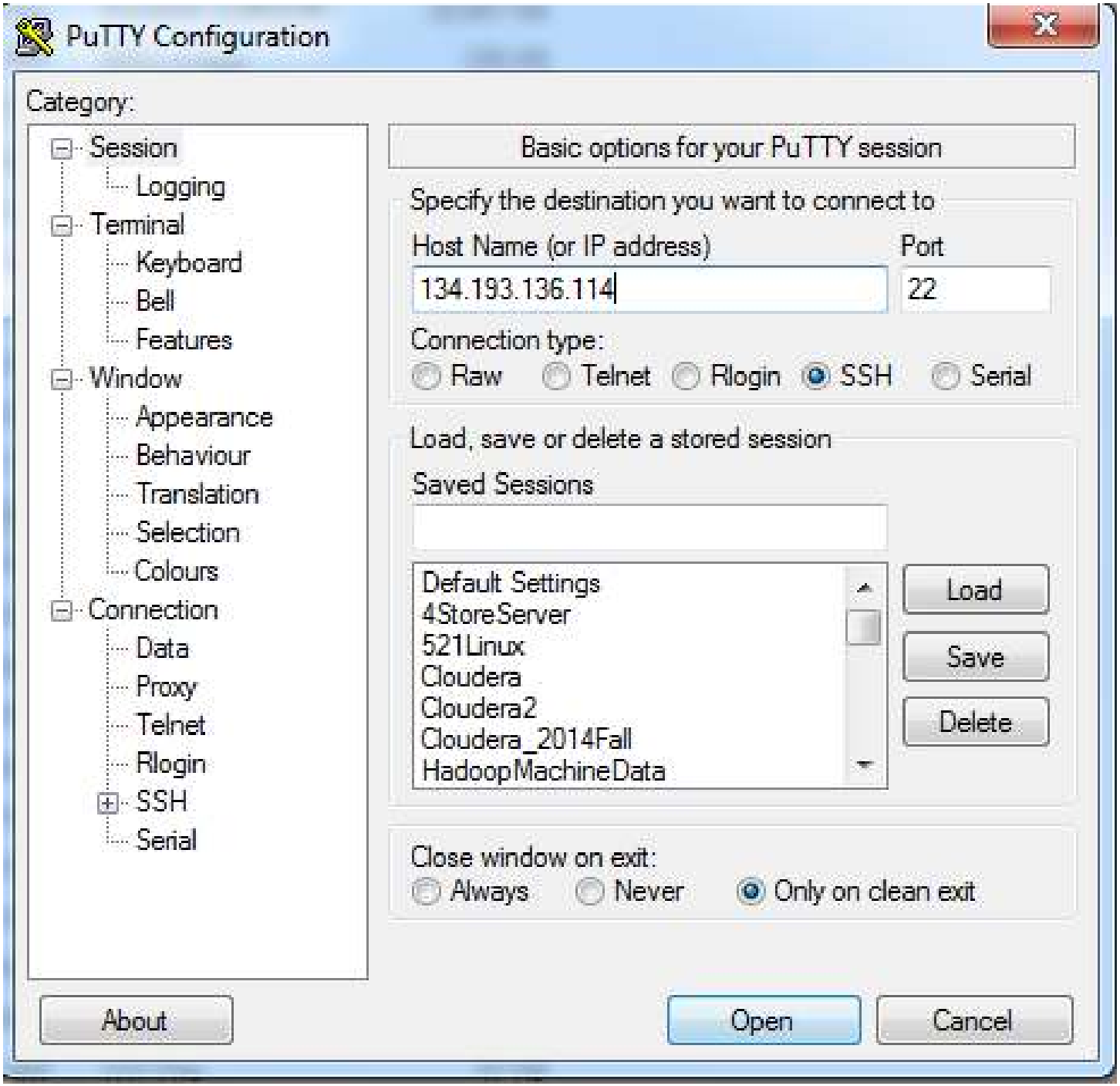
Cloudera/MapReduce

Introduction

- Platform for Apache Hadoop and its ecosystem
- CDH (Cloudera Distribution Including Apache Hadoop) is its open source Apache Hadoop distribution

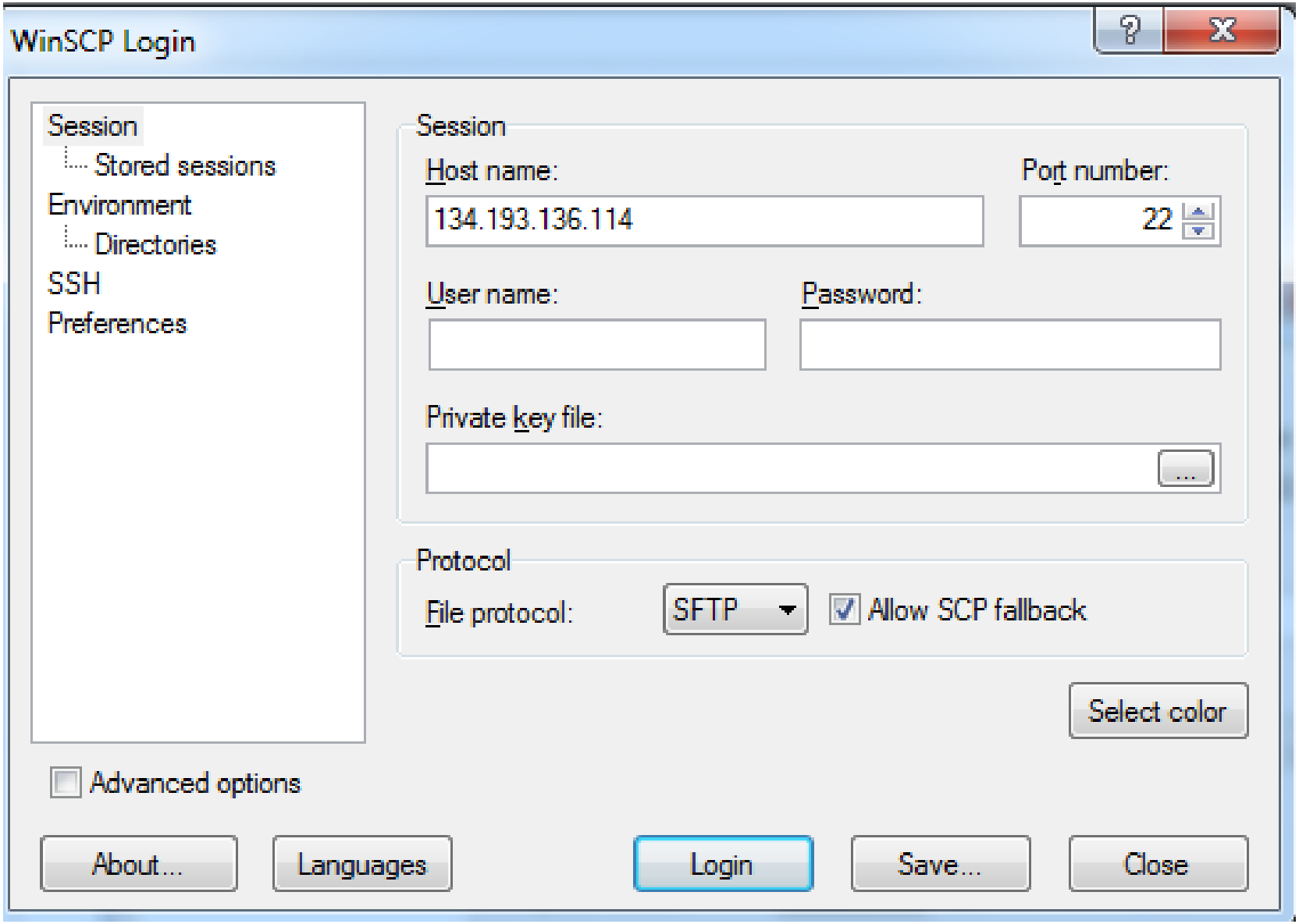
Remote Linux Machine

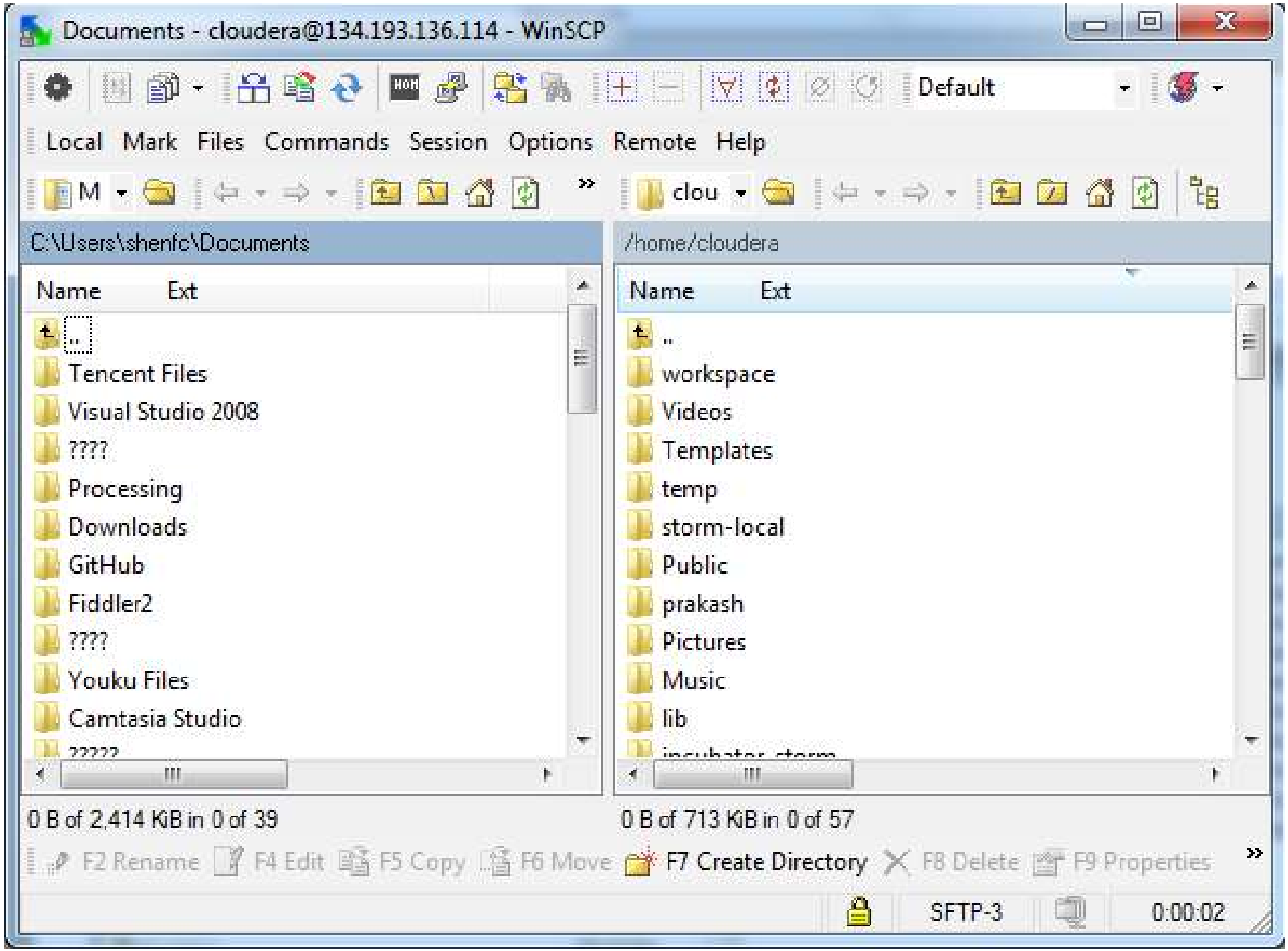
- How to login?
- Download putty http://www.chiark.greenend.org.uk/~sgtatham/putty/
- Host name is 134.193.136.114
- Each group will have an account, user name and password are same
- For example, for group1, user name is *group1*, password is also *group1*
- You can change your password by typing *passwd*


PuTTY Configuration
Category:
Session
Logging
Terminal
Keyboard
Bell
Features
Window
Appearance
Behaviour
Translation
Selection
Colours
Connection
Data
Proxy
Telnet
Rlogin
SSH
Serial
Basic options for your PuTTY session
Specify the destination you want to connect to
Host Name (or IP address)
Port
134.193.136.114
22
Connection type:
Raw
Telnet
Rlogin
SSH
Serial
Load, save or delete a stored session
Saved Sessions
Default Settings
4StoreServer
521Linux
Cloudera
Cloudera2
Cloudera_2014Fall
HadoopMachineData
Load
Save
Delete
Close window on exit:
Always
Never
Only on clean exit
About
Open
Cancel

SSH & FTP tools

- WinSCP provids a very easy way to transfer files remotely from local windows machine to remote linux machine
- http://winscp.net/eng/download.php


WinSCP Login
Session
Stored sessions
Environment
Directories
SSH
Preferences
Session
Host name:
134.193.136.114
Port number:
22
User name:
Password:
Private key file:
...
Protocol
File protocol:
SFTP
Allow SCP fallback
Select color
Advanced options
About...
Languages
Login
Save...
Close


Documents - cloudera@134.193.136.114 - WinSCP
Default
Local Mark Files Commands Session Options Remote Help
M
clou
C:\Users\shenfc\Documents
/home/cloudera
Name Ext
..
Tencent Files
Visual Studio 2008
????
Processing
Downloads
GitHub
Fiddler2
????
Youku Files
Camtasia Studio
Name Ext
..
workspace
Videos
Templates
temp
storm-local
Public
prakash
Pictures
Music
lib
0 B of 2,414 KiB in 0 of 39
0 B of 713 KiB in 0 of 57
F2 Rename F4 Edit F5 Copy F6 Move F7 Create Directory F8 Delete F9 Properties
SFTP-3
0:00:02

SSH & FTP tools

- For Linux and Mac users, Filezilla is also a good choice. You can download it from https://filezilla-project.org/download.php?type=client
- How to use Filezilla:

https://www.youtube.com/watch?v=MCGctsHKizw

1. Run WordCount

- Basic Hadoop command:

Hadoop fs –ls (list everything under HDFS)

Hadoop fs –rm xxx (remove file from HDFS)

Hadoop fs –rm –r xxx (remove dir from HDFS)

Hadoop fs –put xxx xxx (put file from local to HDFS)

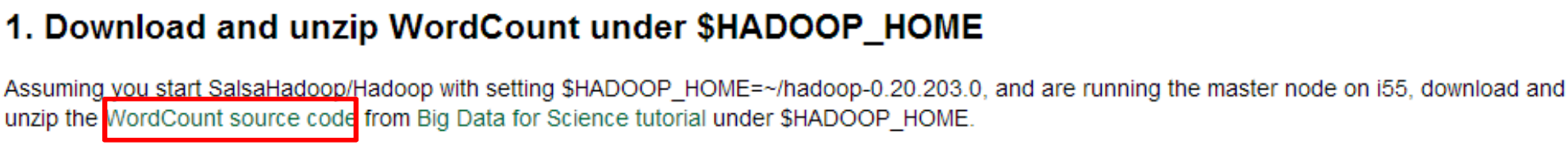
Hadoop fs –cat xxx (concatenate message to screen)

Hadoop jar xxx.jar (run jar file in Hadoop)

1. Run WordCount

- 1. Download wordcount example from this link.

https://portal.futuregrid.org/manual/hadoop-wordcount



1. Download and unzip WordCount under $HADOOP_HOME

Assuming you start SalsaHadoop/Hadoop with setting $HADOOP_HOME=~/hadoop-0.20.203.0, and are running the master node on i55, download and unzip the WordCount source code from Big Data for Science tutorial under $HADOOP_HOME.

- 2. Use FTP tool to transfer the project

1. Run WordCount

- 3. Unzip the wordcount.zip (unzip xxx.zip)
- 4. Go to wordcount folder
- 5. put local input file to the hadoop input directory:

  hadoop fs –put LocalInput HDFSInput
- 6. Run hadoop:

hadoop jar wordcount.jar WordCount input output3

- 7. View result from output3

hadoop fs -cat output3/*

2. Make a hadoop jar and run it in Cloudera

- Instead of running hadoop command *hadoop jar wordcount.jar WordCount input output3*

we can write a jar file to complete the same job and specify input and ouput dir in the java code

- To finish the job, you have to include hadoop library:
  - hadoop-core-1.0.0.jar

2. Make a hadoop jar and run it in Cloudera

```
public class WordCount {
public static void main(String[] args) {
JobClient client = new JobClient();
JobConf conf = new JobConf(WordCount.class);

FileInputFormat.addInputPath(conf, new Path("input"));/// The dir here is the Hadoop directory
FileOutputFormat.setOutputPath(conf, new Path("out7"));
conf.setOutputKeyClass(Text.class);
conf.setOutputValueClass(LongWritable.class);

conf.setMapperClass(TokenCountMapper.class);
conf.setCombinerClass(LongSumReducer.class);
conf.setReducerClass(LongSumReducer.class);
client.setConf(conf);
try {
JobClient.runJob(conf);
} catch (Exception e) {
e.printStackTrace();
}
}
}
```

Hadoop map/reduce algorithm

- Map:

```
String line = value.toString();
List[];
///when met a space, consider word count++
String Arr[] = line.split (“ ”);
For (int i=0;i<Arr.length;i++)
{
    String word = Arr[i];
    List.put(word, 1);
}
```

I	1
a	1
a	1
m	1
a	1
n	1
d	1

Hadoop map/reduce algorithm

- Reduce:

int sum = 0;

I	1
a	3
m	1
n	1
d	1

/// Iterator the hashmap, calculate the total number of a word

for (IntWritable val : values) { sum += val.get(); }
context.write(key, new IntWritable(sum));

2. Make a hadoop jar and run it in Cloudera

- 1). Import the WordCountExercise project into the Eclipse in your laptop

File->import->Existing Projects into Workspace-> Choose “WordCountExercise” project

- 2). Export the project as jar file

2. Make a hadoop jar and run it in Cloudera

- 3). How to export:

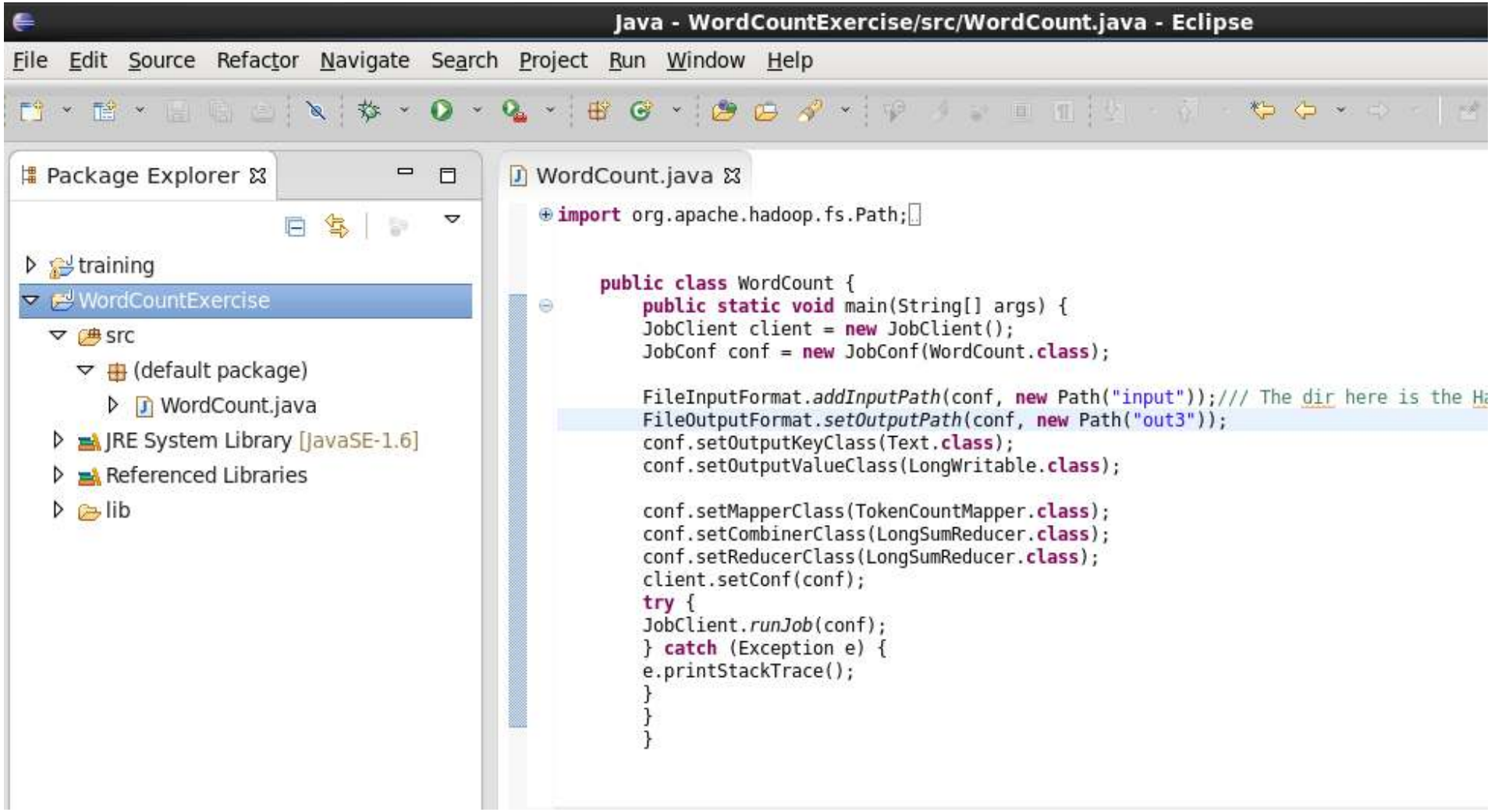
a. Right click project

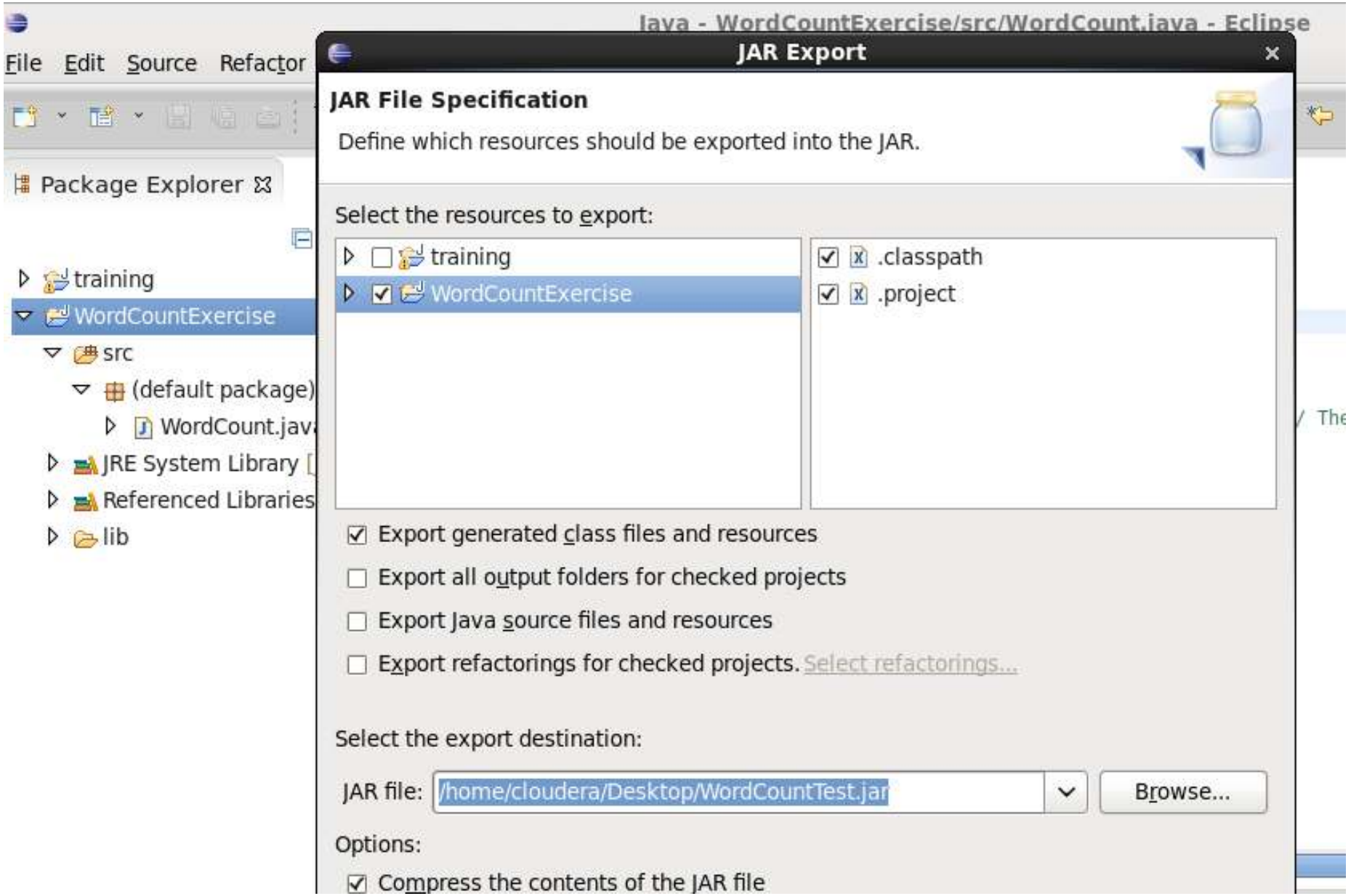
b. Click export

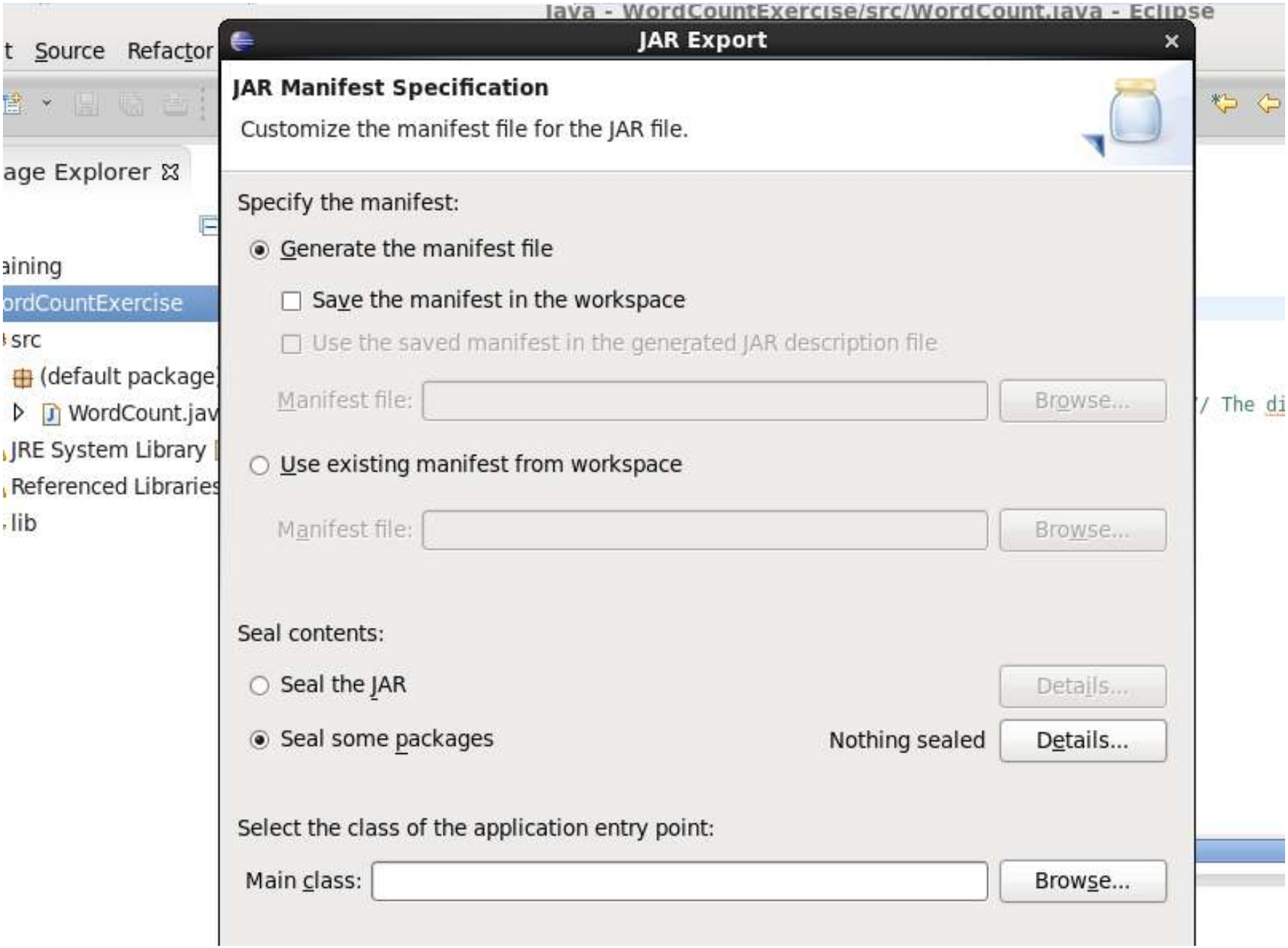
c. Select JAR file

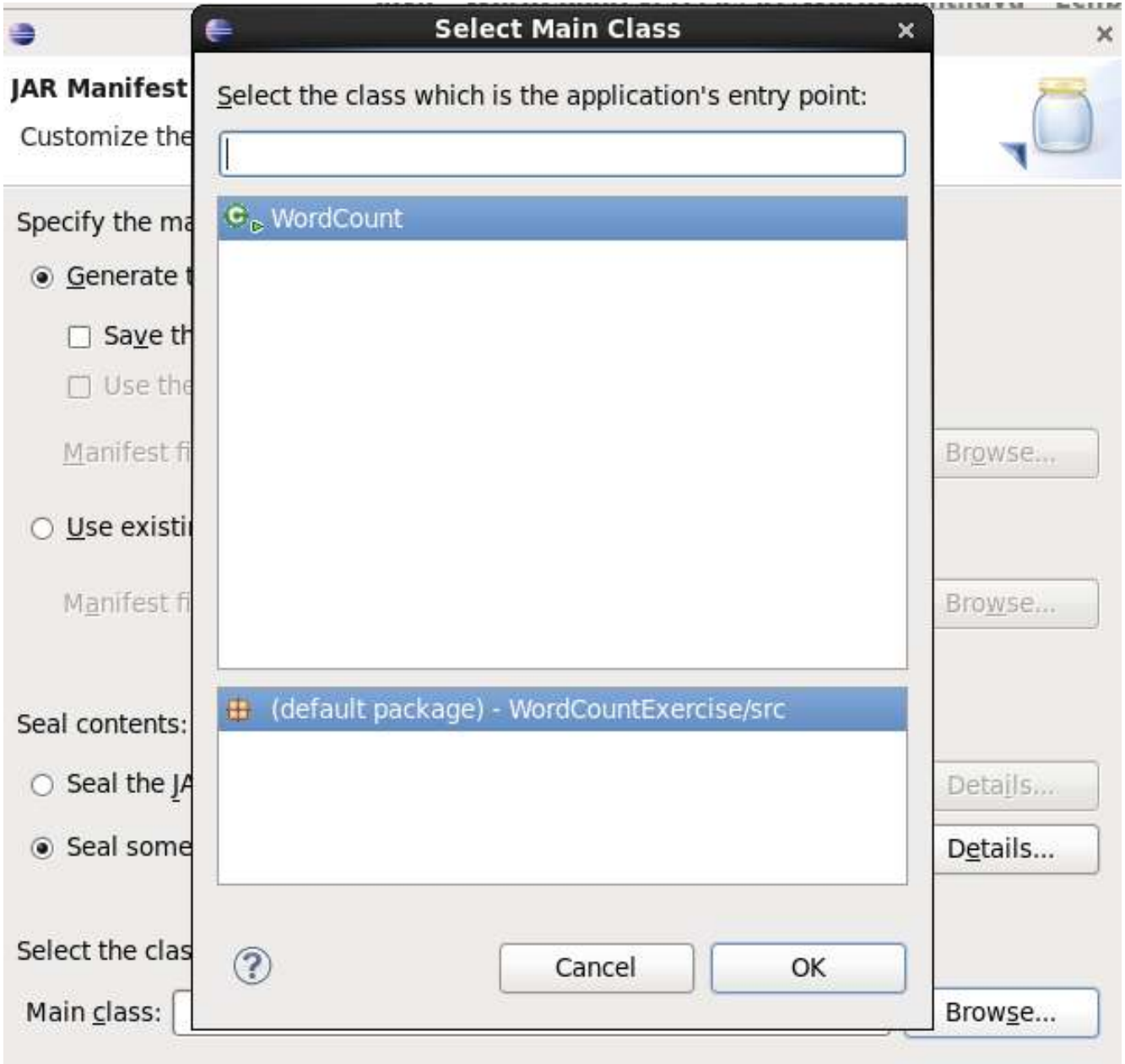
d. Click next, next

e. Don’t forget to select main class by clicking browse to select the main class you want to select.


Java - WordCountExercise/src/WordCount.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
training
WordCountExercise
src
(default package)
WordCount.java
JRE System Library [JavaSE-1.6]
Referenced Libraries
lib
WordCount.java
import org.apache.hadoop.fs.Path;
public class WordCount {
public static void main(String[] args) {
JobClient client = new JobClient();
JobConf conf = new JobConf(WordCount.class);
FileInputFormat.addInputPath(conf, new Path("input"));/// The dir here is the Ha
FileOutputFormat.setOutputPath(conf, new Path("out3"));
conf.setOutputKeyClass(Text.class);
conf.setOutputValueClass(LongWritable.class);
conf.setMapperClass(TokenCountMapper.class);
conf.setCombinerClass(LongSumReducer.class);
conf.setReducerClass(LongSumReducer.class);
client.setConf(conf);
try {
JobClient.runJob(conf);
} catch (Exception e) {
e.printStackTrace();
}
}
}


Java - WordCountExercise/src/WordCount.java - Eclipse
File Edit Source Refactor
Package Explorer
training
WordCountExercise
src
(default package)
WordCount.java
JRE System Library
Referenced Libraries
lib
JAR Export
JAR File Specification
Define which resources should be exported into the JAR.
Select the resources to export:
training
WordCountExercise
.classpath
.project
Export generated class files and resources
Export all output folders for checked projects
Export Java source files and resources
Export refactorings for checked projects. Select refactorings...
Select the export destination:
JAR file: /home/cloudera/Desktop/WordCountTest.jar
Browse...
Options:
Compress the contents of the JAR file


Java - WordCountExercise/src/WordCount.java - Eclipse
t Source Refactor
age Explorer
aining
ordCountExercise
src
(default package)
WordCount.jav
JRE System Library
Referenced Libraries
lib
JAR Export
JAR Manifest Specification
Customize the manifest file for the JAR file.
Specify the manifest:
Generate the manifest file
Save the manifest in the workspace
Use the saved manifest in the generated JAR description file
Manifest file:
Browse...
Use existing manifest from workspace
Manifest file:
Browse...
Seal contents:
Seal the JAR
Details...
Seal some packages
Nothing sealed
Details...
Select the class of the application entry point:
Main class:
Browse...
/ The di


Select Main Class
Select the class which is the application's entry point:
WordCount
(default package) - WordCountExercise/src
Cancel
OK
JAR Manifest
Customize the
Specify the ma
Generate t
Save th
Use the
Manifest fi
Browse...
Use existi
Manifest fi
Browse...
Seal contents:
Seal the JA
Details...
Seal some
Details...
Select the clas
Main class:
Browse...

2. Make a hadoop jar and run it in Cloudera

- 4). Run jar by the command:

hadoop jar WordCountExercise.jar

- 5). Check result:

hadoop fs –cat out3/*

Tips

- If you want to run this code again, in java code, you must change the output directory to another new value, like out4 and out5, and export to jar again:

❑ FileOutputFormat.*setOutputPath(conf,* ***new Path("out4"));***

Otherwise, hadoop will report the error like: output directory already exists.

Thanks