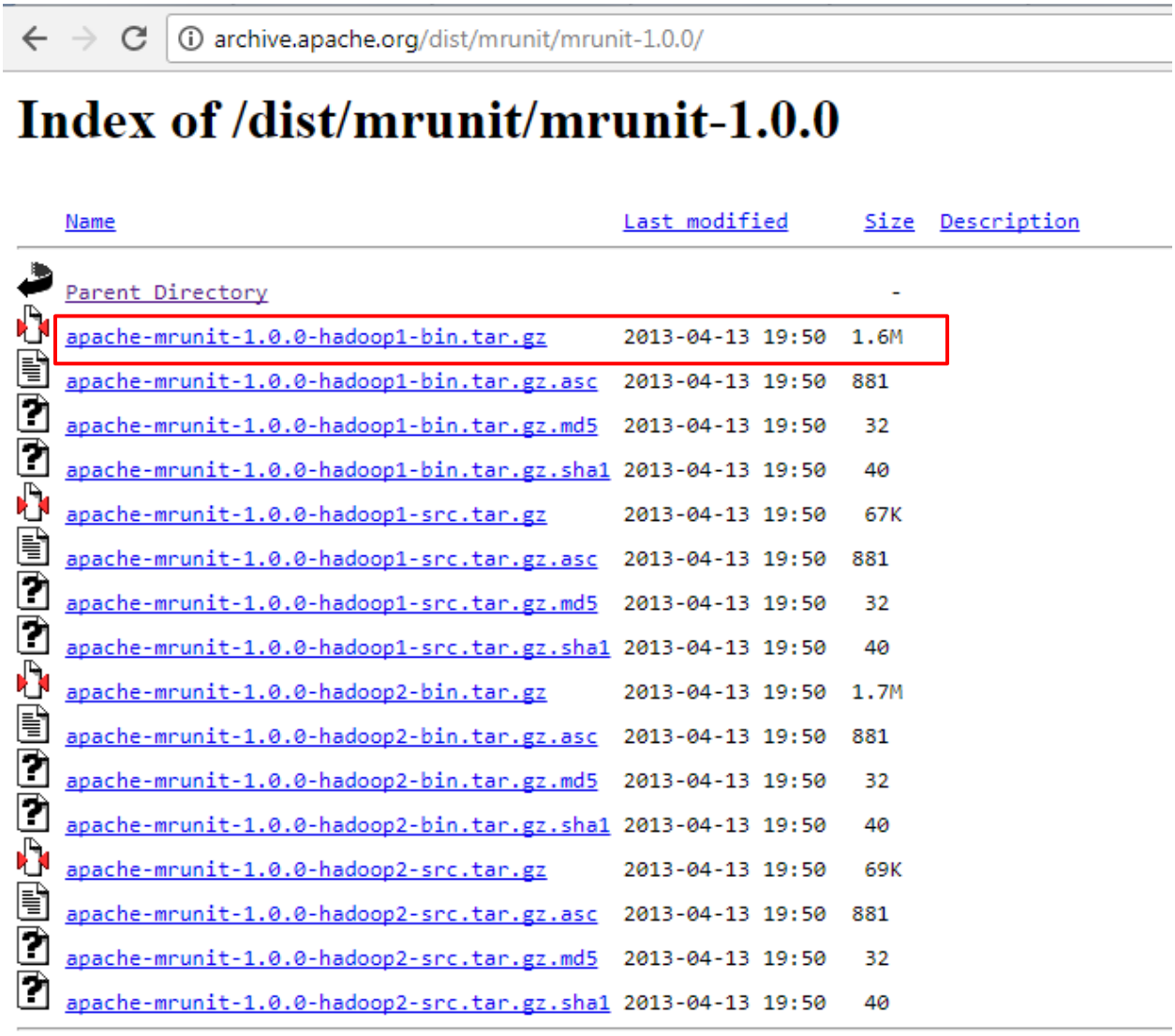
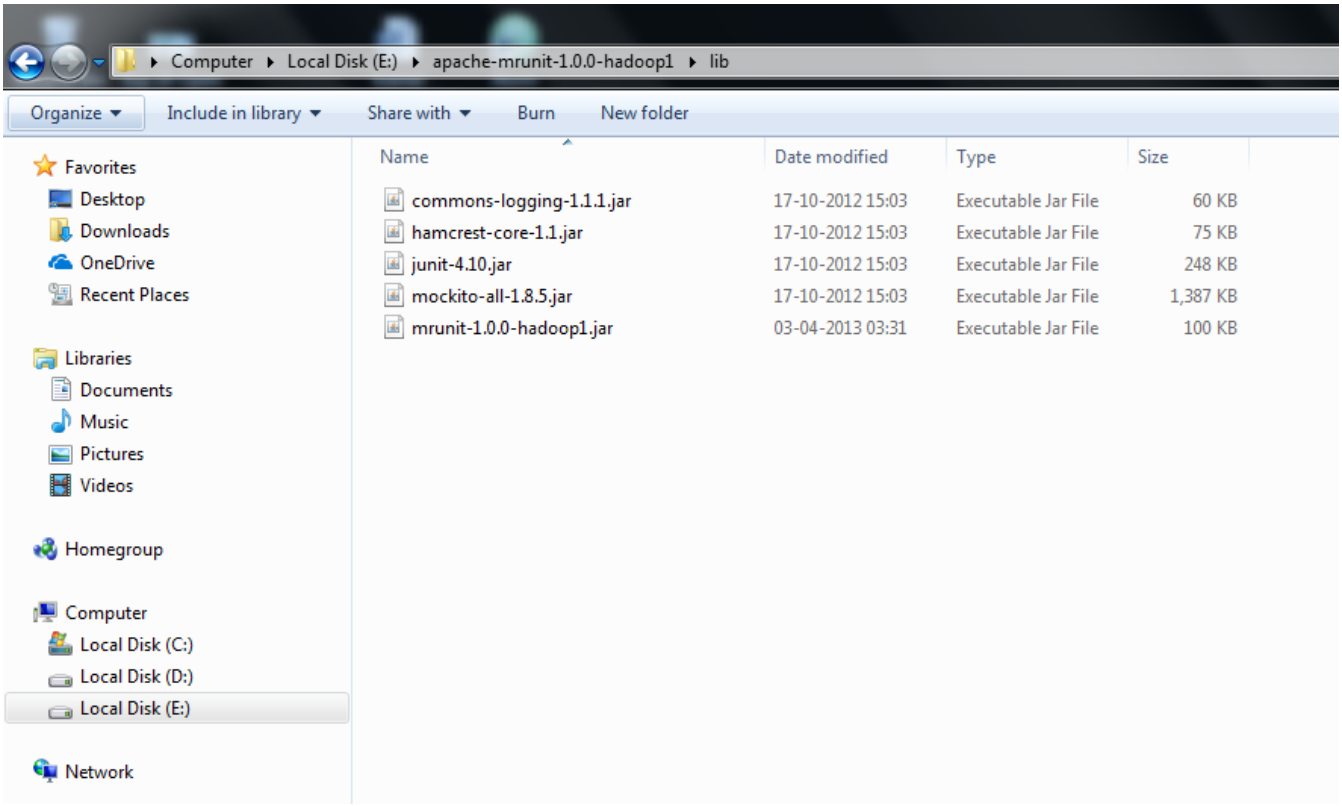
MRUnit Testing

1.Download **apache-mrunit-1.0.0-hadoop1** jars which contains the Mrunit testing api using link given below:

http://archive.apache.org/dist/mrunit/mrunit-1.0.0/

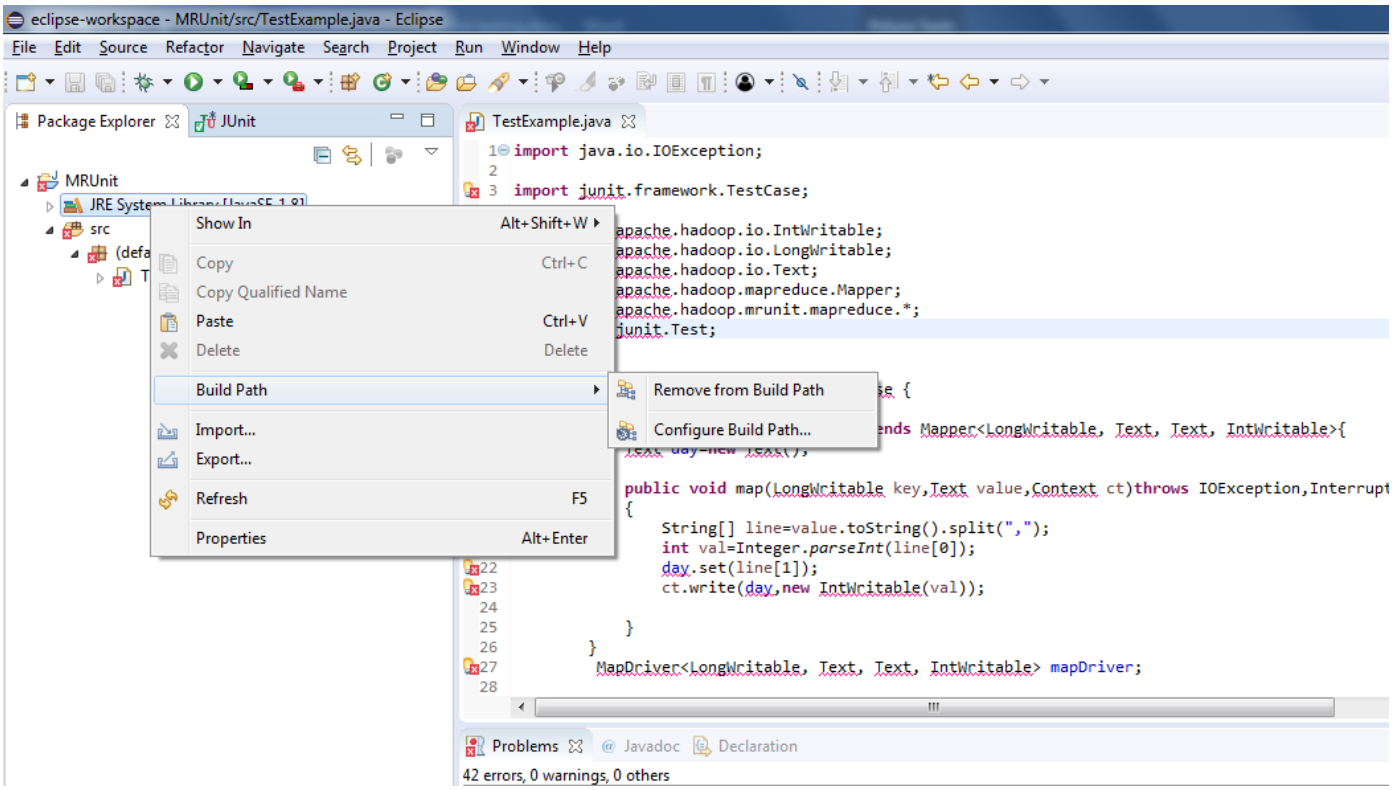



Computer ▸ Local Disk (E:) ▸ apache-mrunit-1.0.0-hadoop1 ▸ lib
Organize
Include in library
Share with
Burn
New folder
Favorites
Desktop
Downloads
OneDrive
Recent Places
Libraries
Documents
Music
Pictures
Videos
Homegroup
Computer
Local Disk (C:)
Local Disk (D:)
Local Disk (E:)
Network
Name
Date modified
Type
Size
commons-logging-1.1.1.jar 17-10-2012 15:03 Executable Jar File 60 KB
hamcrest-core-1.1.jar 17-10-2012 15:03 Executable Jar File 75 KB
junit-4.10.jar 17-10-2012 15:03 Executable Jar File 248 KB
mockito-all-1.8.5.jar 17-10-2012 15:03 Executable Jar File 1,387 KB
mrunit-1.0.0-hadoop1.jar 03-04-2013 03:31 Executable Jar File 100 KB

2. Import all jars from the lib folder of the downloaded file into eclipse project.

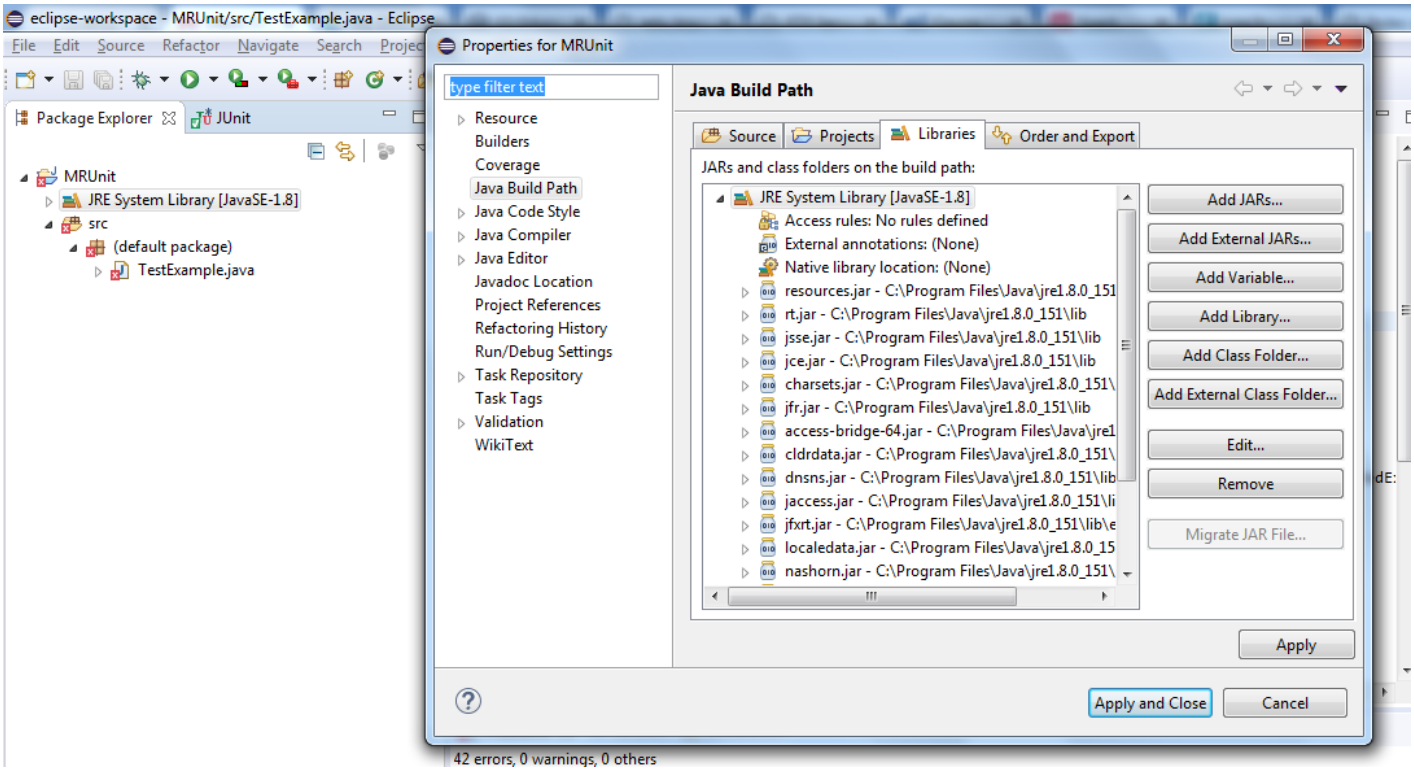
a. Right click on JRE system library

b. Select Build Path

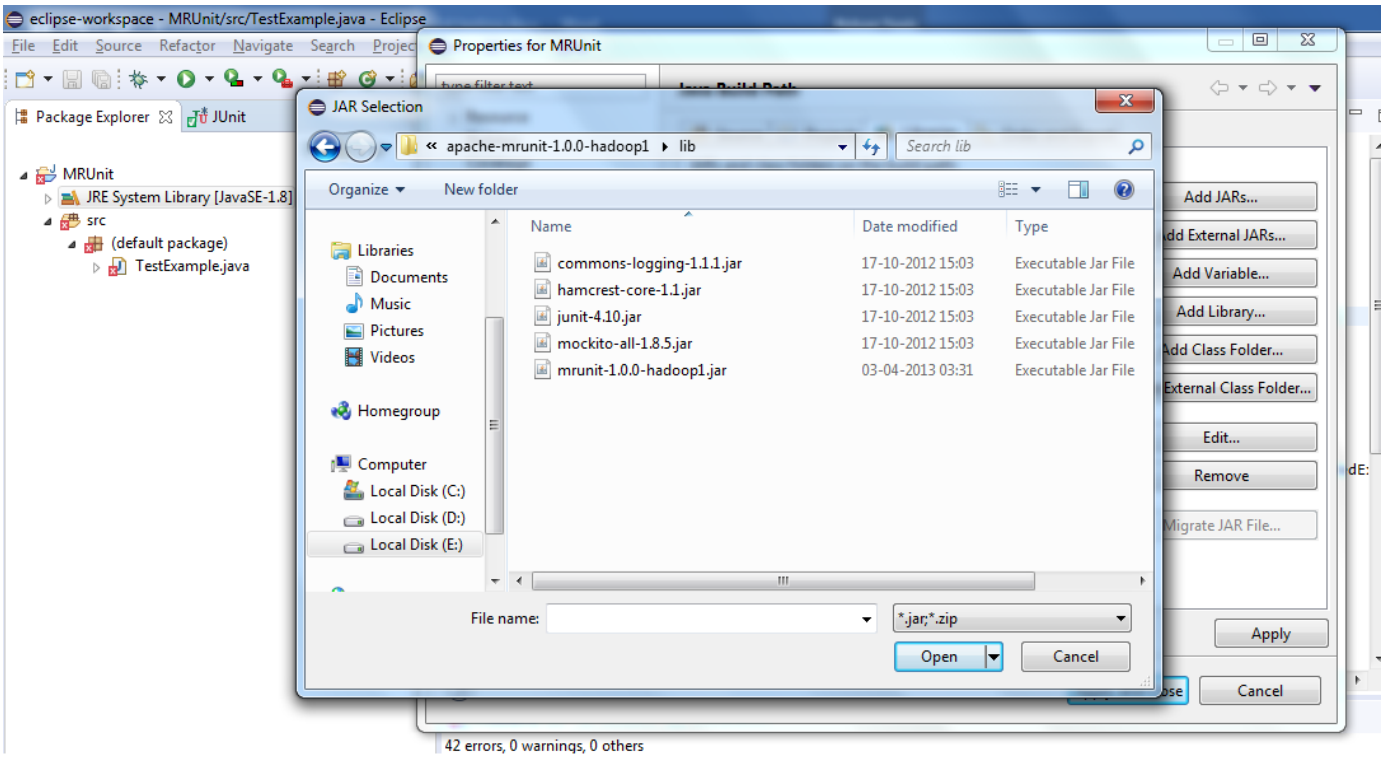


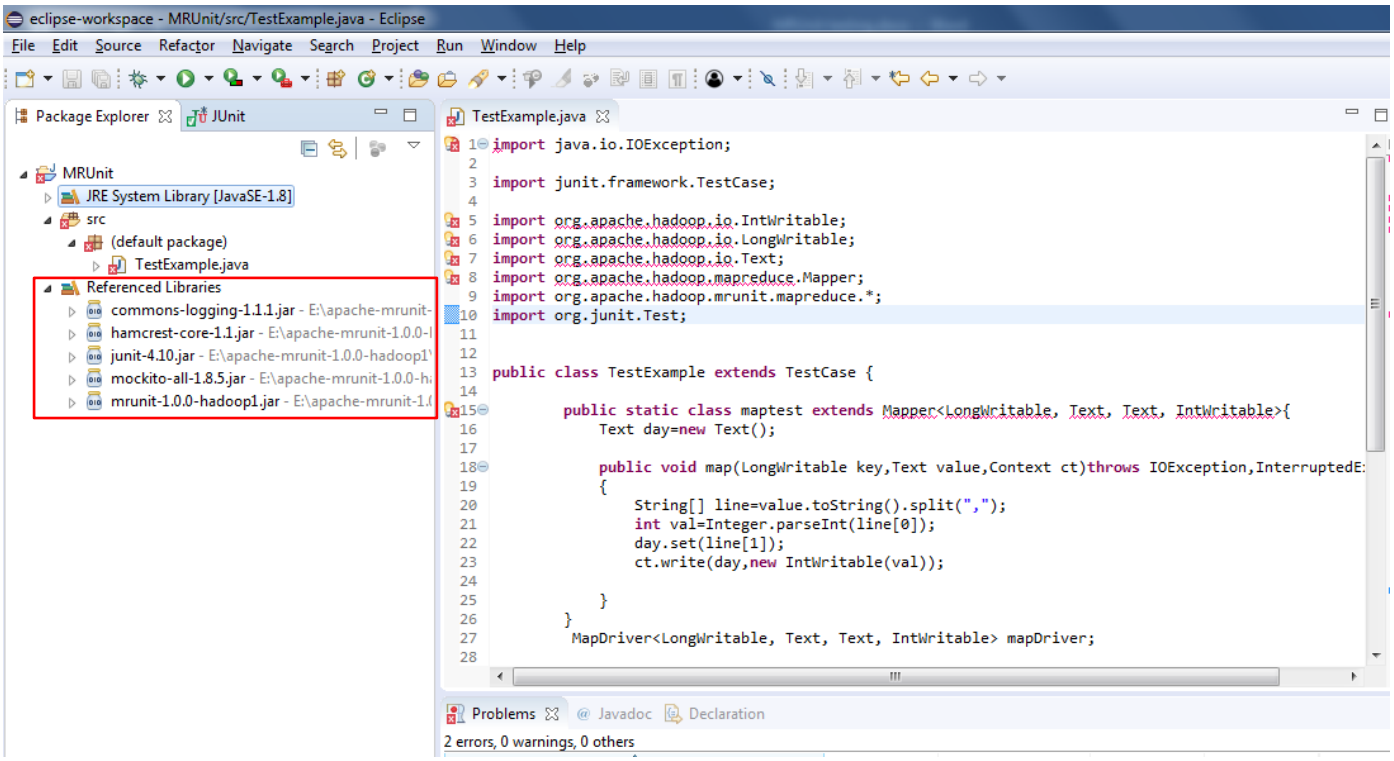
c. Select configure build path

d. Select *Add External Jars* and browse the lib folder where you have downloaded the apache-mrunit-1.0.0-hadoop1 file.



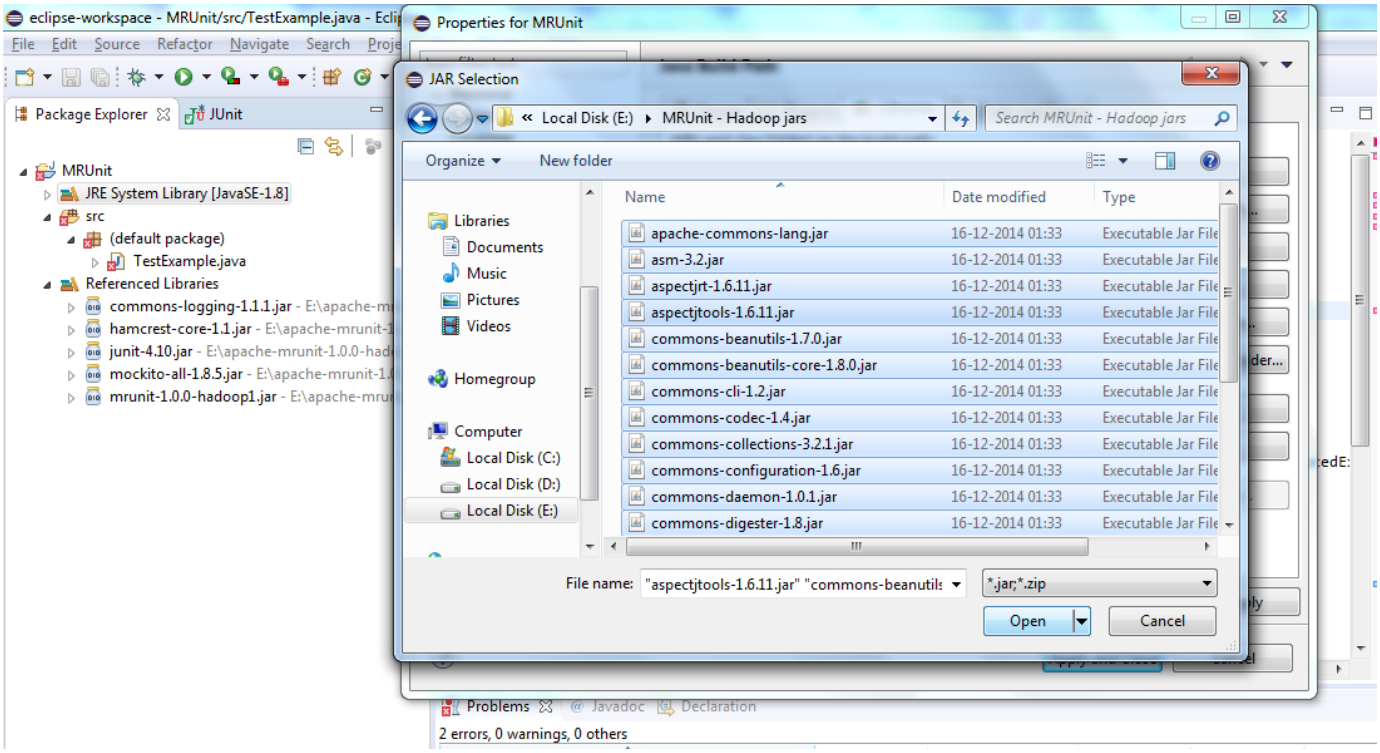
e. Select all jars to add them to your testing project.





3. Download all Hadoop jars using link given below:
https://goo.gl/wcVoHV

4. Import all the downloaded Hadoop jars into eclipse project in the same manner as shown in step 2.



Testing Mappers

1. Instantiate an instance of the MapDriver class parameterized exactly as the mapper under test.
2. Add an instance of the Mapper you are testing in the withMapper call
3. In the withInput call pass in your key and input value,here LongWritable with an arbitrary value and a Text object that contains a line “First day,Sunday,abhay,holiday”.
4. Specify the expected output in the withOutput call,here we are expecting a Text object as the key with value as “Sunday” and the count of it as the value as “1”.
5. The last call runTest feeds the specified input values into the mapper and compares the actual output against the expected output set in the ‘withOutput’ method.

Note: *The MapDriver only allows one input and output per test. You can call withInput and withOutput multiple times if you want, but the MapDriver will overwrite the existing values with the new ones, so you will only ever be testing with one input/output at any time.*

The program is given below:

```
import java.io.IOException;

import junit.framework.TestCase;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Mapper;

import org.apache.hadoop.mrunit.mapreduce.*;

import org.junit.Test;


public class TestExample extends TestCase {

public static class maptest extends Mapper<LongWritable, Text, Text, IntWritable>{

                    Text day=new Text();

public void map(LongWritable key,Text value,Context ct)throws
IOException,InterruptedException

                    {

                          String[] line=value.toString().split(",");

                          int val=Integer.parseInt(line[0]);

                          day.set(line[1]);

                          ct.write(day,new IntWritable(val));


                    }

              }

               MapDriver<LongWritable, Text, Text, IntWritable> mapDriver;
```

```
 public void setUp() {
   new maptest();
    mapDriver = MapDriver.newMapDriver( new maptest());
  }
  public void testMapper() {
    try {
mapDriver.withInput(new LongWritable(), new Text("First day,Sunday,abhay,holiday"))
                      .withOutput(new Text("sunday"), new IntWritable(1))
                      .runTest();
      } catch (IOException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
      }
  }
}
```