一

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import jdk.nashorn.internal.ir.CatchNode;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.fs.FSDataInputStream;

import org.apache.hadoop.fs.FileSystem;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.hdfs.server.namenode.FSAclBaseTest;

public class MergeFile {

public static void main(String[] args)

{

try{

String filename = "MyFile.txt";

Path filePath=new Path(filename);

Configuration conf = new Configuration();

conf.set("fs.defaultFS", "hdfs://localhost:9000");

conf.set("fs.hdfs.impl","org.apache.hadoop.hdfs.DistributedFileSystem");

FileSystem fs = FileSystem.get(conf);

if(fs.exists(filePath)){

FSDataInputStream getIt = fs.open(filePath);

BufferedReader d = new BufferedReader(new InputStreamReader(getIt));

String str = null;

while ((str = d.readLine())!=null) {

System.out.println(str);

}

d.close();

}else {

System.out.println("文件不存在！！！！！！");

}

fs.close();

} catch(IllegalArgumentException e){

e.printStackTrace();

}catch(IOException e) {

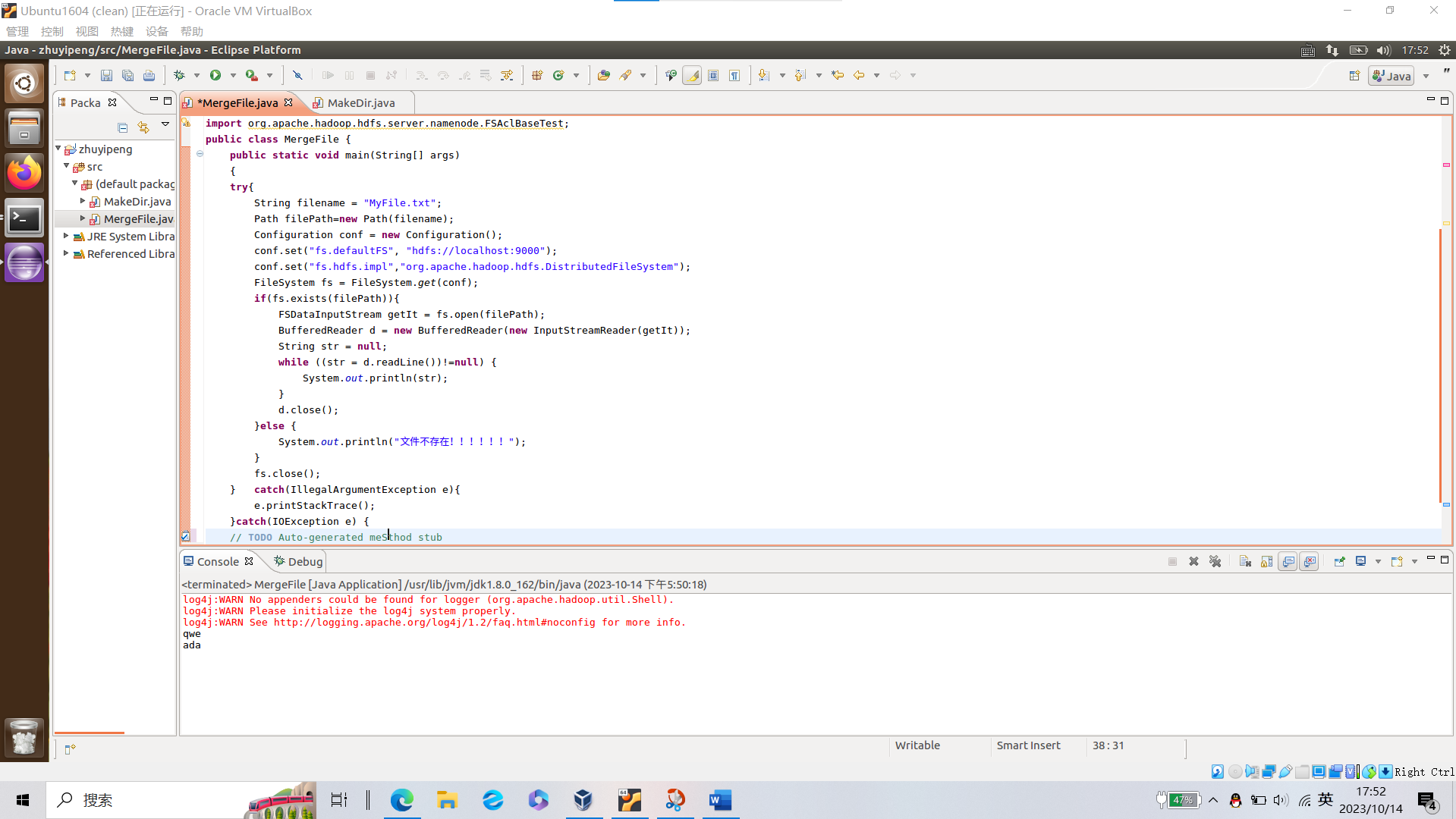
// TODO Auto-generated method stub

e.printStackTrace();

}

}

}



二

import java.io.FileNotFoundException;

import java.io.IOException;

import java.io.InputStreamReader;

import java.io.PrintStream;

import java.net.URI;

import org.apache.hadoop.classification.InterfaceAudience.Private;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.fs.FSDataInputStream;

import org.apache.hadoop.fs.FSDataOutputStream;

import org.apache.hadoop.fs.FileStatus;

import org.apache.hadoop.fs.FileSystem;

import org.apache.hadoop.fs.Path;

import org.apache.yetus.audience.InterfaceAudience.Public;

import org.checkerframework.common.value.qual.StaticallyExecutable;

public class MakeDir {

private URI uri;

public static FileSystem fs;

private static Configuration conf;

private static Path path;

public static void set()throws IOException

{

conf = new Configuration();

conf.set("fs.defaultFS","hdfs://localhost:9000");

conf.set("fs.hdfs.impl","org.apache.hadoop.hdfs.DistributedFileSystem");

fs = FileSystem.get(conf);

path=new Path("/hdfsjava");

}

//创建目录

public static void createdir()throws IOException

{

if(fs.mkdirs(path))

System.out.println("创建目录成功");

else

System.out.println("创建目录失败");

}

//创建文件

public static void createfile()throws IOException

{

path = new Path("/hdfsjava/mobile.txt");

FSDataOutputStream output = fs.create(path);

output.write("My telePhone is HUAWEI".getBytes());

output.flush();

output.close();

}

//查看文件内容

public static void catfile()throws IOException

{

FSDataInputStream input =fs.open(path);

byte[]data =new byte[1024];

int read=-1;

PrintStream ps =new PrintStream(System.out);

while((read=input.read(data))!=-1)

{

ps.write(data,0, read);

}

//ps.close();

//input.close();

}

//上传文件

public static void copyfromlocalfile()throws IOException

{

Path srcPath=new Path("/home/hadoop/myLocalFile.txt");

Path dstPath=new Path("/hdfsjava/");

fs.copyFromLocalFile(srcPath, dstPath);

}

//查看hdfsjava下所有文件

@SuppressWarnings("deprecation")

public static void FileList()throws FileNotFoundException,IOException

{

Path path2 = new Path("/hdfsjava/");

FileStatus[]list = fs.listStatus(path2);

for (FileStatus f : list) {

System.out.printf("name:%s,folder:%s,size:%d\n",f.getPath(),

f.isDir(),f.getLen());

}

}

public static void copyfromhdfsfile()throws IOException

{

Path destpath2=new Path("/home/stu");

Path srcPath2=new Path("/hdfsjava/mobile.txt");

fs.copyToLocalFile(srcPath2, destpath2);

}

public static void deletefile()throws IOException

{

boolean b = fs.delete(path,true);

if(b)

System.out.println("删除成功！");

}

public static void main(String[]args)

{

try {

set();

createdir();

createfile();

copyfromlocalfile();

FileList();

catfile();

copyfromhdfsfile();S

deletefile();

} catch (Exception e) {

// TODO: handle exception

e.printStackTrace();

}

}

}

