

package com.hlj.file;

import java.io.BufferedReader;

import java.io.ByteArrayInputStream;

import java.io.ByteArrayOutputStream;

import java.io.File;

import java.io.FileInputStream;

import java.io.FileNotFoundException;

import java.io.FileOutputStream;

import java.io.IOException;

import java.io.InputStreamReader;

import java.io.StringWriter;

import java.io.UnsupportedEncodingException;

import java.net.MalformedURLException;

import java.net.URL;

import java.text.SimpleDateFormat;

import java.util.Date;

import java.util.Random;

import org.dom4j.Document;

import org.dom4j.DocumentHelper;

import org.dom4j.io.OutputFormat;

import org.dom4j.io.XMLWriter;

public class FileTest {

/\*\*

\* 6、

\*/

static int i = 1; //用于保留统计记录的个数

public static void main(String[] args) throws Exception {

/\*\*

\* 1、已知路径创建 相关文件 前提，必须存在路径 ，没有路径会报错

\*/

// createFile();

/\*\*

\* 1.2、创建有父亲目录的文件

\*

\*/

// createHavaDirectoryFile();

/\*\*

\* 2、判断是不是文件或者目录

\* 如果上述文件、目录不存在，也不会报错

\*/

// File directoryName = new File("D:/test/healerjean/txt.txt");

// judgeFileOrDirectory(directoryName);

/\*\*

\* 3、

\* 判断是不是目录

\* 创建目录，即使路径不存在，也会创建相关路径，因为是mkdirs

\*/

// createDirectory();

/\*\*

\* 4、复制文件

\* 第一个文件以及路径必须存在， 否则fileInputStream会报错

\* 第二个文件可以不存在,但是路径必须存在，如果路径不存在则FileOutSteam会报错

\*/

copyFile("D:/test/healerjean/file.txt","D:/test/copyFile/file2.txt");

/\*\*

\* 5、删除目录

\*

\*/

// File file = new File("C:/Users/qdkf/Desktop/项目/各省社保整理/ceshi");

// delteFiles(file);

/\*\*

\* 6 、目录下读取文件内容进行匹配

\*

\*/

//File file = new File("C:\\Users\\qdkf\\Desktop\\项目\\各省社保整理\\数据包\\数据包\\福建社保--人保联调数据包");

// File file = new File("E:/study/HealerJean123.github.io");

// getFileValue(file,"asidetitle");

/\*\*

\* 7、获取txt文件内容

\*

\*/

// String filePath = "D:\\test\\20170904\\9108\_REQ\_XML\_T0120170904006463.xml";

// getFileTextValue(filePath);

/\*\*

\* 8、按照行,一行一行读取内容

\*/

// String filePath = "C:/Users/qdkf/Desktop/项目/江苏/STS01ToRST01\_NJF01001\_20170904\_2017090465853776438363389189.txt";

// getFileReadLineTextValue(filePath);

/\*\*

\* 9、根据字符串生成内容

\* 只要修改code city testType typecChoice directChoice

\*/

/\*String textContext = XmlTestContet.jiangSu1104RsToStEnd;

String code = "1104-"; //修改位置 1

String city = "江苏-"; //修改位置 2

String testType = getTestType("1104").trim(); //修改位置 3

String typeOne = "发送报文-";

String typeTwo = "响应报文-";

int typecChoice = 0; //1发送报文 //修改位置 4

String typecChoiceFinal = null;

if(typecChoice==1){

typecChoiceFinal = typeOne;

}else{

typecChoiceFinal = typeTwo;

}

String directRS = "代理人发受托人-";

String directST = "受托人发代理人-";

int directChoice = 1; //1 代理人发受托人 //修改位置 5

String directChoiceFinal = null;

if(directChoice==1){

directChoiceFinal = directRS;

}else {

directChoiceFinal = directST;

}

SimpleDateFormat simpleDateFormat = new SimpleDateFormat("yyyyMMdd-HHmmss");

String ds =simpleDateFormat.format(new Date());

String fileName = city+typecChoiceFinal+code+ds+".xml";

//例如

String path = "D:/txtDirectory/"+city+"/"+testType+"/"+code+directChoiceFinal+"/"+typecChoiceFinal;

getTxtByTextContent(textContext,fileName,path);

\*/

/\*\*

\*10、 获取当前 操作系统信息 以及路径问题

\*/

// testSeparator();

}

public static String getTestType(String str){

if(str.equals("0101")||str.equals("SYS00101")||str.equals("SYS02000")){

return "连通性测试";

}if(str.equals("0102")||str.equals("SYS01090")||str.equals("SYS02090")){

return "通用交易查询";

}if(str .equals("1101")||str.equals("NJB02101")){

return "计划基本信息下载";

}if(str .equals("1102")||str.equals("NJB02103")){

return "投资组合信息推送";

}if(str .equals("1103")||str.equals("NJB02102")){

return "年金管理人信息推送";

}if(str .equals("1104")||str.equals("NJB02206")){

return "定价日信息推送";

}if(str .equals("1201")||str.equals("NJB01201")){

return "划款指令上传";

}if(str .equals("1203")||str.equals("NJB02201")){

return "划款到账回执";

}if(str .equals("1210")||str.equals("NJB02202")){

return "单位净值信息上传";

}if(str .equals("1211")||str.equals("NJB01202")){

return "资产估值明细下载";

}if(str .equals("1212")||str.equals("NJB01203")){

return "成交汇总报表上传";

}if(str .equals("1205")||str.equals("NJB01901")){

return "文件就绪通知(发放通知)";

}if(str .equals("1208")||str.equals("NJB01902")){

return "文件就绪通知(发放结果反馈通知)";

}if(str .equals("9208")||str.equals("NJB02901")||str.equals("NJB02902")){

return "文件处理结果通知";

}

return null;

}

/\*\*

\* 1、

\* 已知路径创建 相关文件

\* 前提，必须存在路径,否则创建文件会报错

\* @throws IOException

\*/

private static void createFile() throws IOException {

File fileNameTxt = new File("D:/test/healerjean/file.txt");

if(!fileNameTxt.exists()){

fileNameTxt.createNewFile();

}

System.out.println("测试成功");

}

/\*\*

\* 1.2、创建有目录的文件

\*

\*/

private static void createHavaDirectoryFile() throws IOException {

File directoryName = new File("D:/test/healerjean/file");

//通过父的目录引入文件 a1 是父类 (只要是之前的父类以上都是可以的)

File directoryNameTxt = new File(directoryName,"directoryName.txt");

if(!directoryNameTxt.exists()){

directoryNameTxt.createNewFile();

}

System.out.println("创建有目录的文件成功");

}

/\*\*

\* 2、

\* 判断是不是文件或者目录

\* 同时也不会报错

\*/

private static void judgeFileOrDirectory(File path) {

System.out.println("测试开始");

if(path.isFile()){

System.out.println(path.getPath()+"是一个文件");

}

else if(path.isDirectory()){

System.out.println(path.getPath()+"是一个目录");

}else {

System.out.println(path.getPath()+"不是文件也不是目录");

}

}

/\*\*

\* 3、

\* 判断是不是目录

\* 创建目录，即使路径不存在，也会创建相关路径，因为是mkdirs

\*/

private static void createDirectory() {

//引入目录

File directoryName = new File("D:/test/healerjean/file");

if(!directoryName.exists()){

directoryName.mkdirs();

System.out.println(directoryName.getPath()+"创建目录成功");

}

}

/\*\*

\* 4、复制文件

\* 第一个文件以及路径必须存在， 否则fileInputStream会报错

\* 第二个文件可以不存在,但是路径必须存在，如果路径不存在则FileOutSteam会报错

\*/

public static void copyFile(String string,String string2) throws IOException{

FileInputStream ins = new FileInputStream(string);

FileOutputStream out = new FileOutputStream(string2);

byte[] b = new byte[1024];

int n=0;

while((n=ins.read(b))!=-1){

out.write(b, 0, n);

}

ins.close();

out.close();

System.out.println("复制文件成功");

}

/\*\*

\* 5、删除多个目录

\*/

public static void delteFiles(File file){

//file.listFiles()是获取file这个对象也就是file这个目录下面的文件和文件夹的集合

File[] files=file.listFiles();

for(File f:files)

{

if(f.isDirectory())//递归调用

{

delteFiles(f);

}

else {

f.delete();

}

}

file.delete();

System.out.println("删除目录成功");

}

/\*\*

\* 6 、目录下读取文件内容进行匹配

\* @throws Exception

\*

\*/

private static void getFileValue(File file,String content) throws Exception {

//

//file.listFiles()是获取file这个对象也就是file这个目录下面的文件和文件夹的集合

File[] files=file.listFiles();

for(File f:files)

{

if(f.isDirectory())//递归调用

{

getFileValue(f,content);

}

else {

String path = f.getPath();

String xml = path.substring(path.lastIndexOf(".")+1, path.length());

/\* if(xml.equals("xml")){\*/

FileInputStream fInputStream = new FileInputStream(f);

ByteArrayOutputStream outStream = new ByteArrayOutputStream();

byte[] buffer = new byte[1024];

int len = 0;

while((len=fInputStream.read(buffer))!=-1){

outStream.write(buffer, 0, len);

}

String str = new String(outStream.toByteArray(),"utf-8");

// System.out.println("文件名为"+f.getName()+"/n的内容为"+str);

// if(str.contains("<typecode>9208</typecode>")&&str.contains("<appcode>0000</appcode>")){

// if(str.contains("<typecode>1103</typecode>")&&!str.contains("<appcode>0000</appcode>")){

if(str.contains(content)){

System.out.println("第"+i+"个文件名为\n"+f.getPath());

System.out.println("内容为\n"+str);

i++;

}

/\* } \*/

}

}

}

/\*\*

\* 7、获取txt文件内容

\*

\*/

private static String getFileTextValue(String filePath) throws Exception {

//

FileInputStream fileInputStream = new FileInputStream(filePath);

ByteArrayOutputStream byteOutputStream = new ByteArrayOutputStream();

int len = 0;

byte[] buffer = new byte[1024];

while((len = fileInputStream.read(buffer))!=-1){

byteOutputStream.write(buffer, 0, len);

}

String txtValue = new String(byteOutputStream.toByteArray());

System.out.println(txtValue);

return txtValue;

}

/\*\*

\* 8、按照行,一行一行读取txt内容

\*/

private static String getFileReadLineTextValue(String filePath) throws Exception {

FileInputStream fileInputStream = new FileInputStream(filePath);

ByteArrayOutputStream byteOutputStream = new ByteArrayOutputStream();

int len = 0;

byte[] buffer = new byte[1024];

while((len = fileInputStream.read(buffer))!=-1){

byteOutputStream.write(buffer, 0, len);

}

byte[] txtByteArray = (byteOutputStream.toByteArray());

BufferedReader reader = new BufferedReader(new InputStreamReader(new ByteArrayInputStream(txtByteArray),"utf-8"));

String lineVal =null ;

int flagLength = 0;

while((lineVal = reader.readLine()) != null ){

flagLength++;

if(flagLength==1){

System.out.println("第"+flagLength+"行的内容为"+lineVal);

}

else{

System.out.println("第"+flagLength+"行的内容为"+lineVal);

}

}

return null;

}

/\*\*

\* 9、根据字符串生成内容

\* 只要修改code city testType typecChoice directChoice

\*/

public static void getTxtByTextContent(String textContext,String fileName,String path){

try {

File fileDirectory = new File(path);

if(!fileDirectory.exists()){

fileDirectory.mkdirs();

}

File fileTxt = new File(path+"/"+fileName) ;

if(!fileTxt.exists()){

fileTxt.createNewFile();

}

FileOutputStream outputStream = new FileOutputStream(fileTxt);

byte[] buffer = textContext.getBytes("utf-8");

outputStream.write(buffer);

System.out.println("成功"+path+"/"+fileName);

} catch (FileNotFoundException e) {

//

e.printStackTrace();

}catch (UnsupportedEncodingException e) {

//

e.printStackTrace();

} catch (IOException e) {

//

e.printStackTrace();

}

}

/\*\*

\* 10、操作系统 路径符号判断

\* @author 作者 HealerJean:

\* @version 创建时间：2017年11月1日 上午9:32:49

\*

\*/

public static void testSeparator(){

/\*\*

\* 1、判断哪个系统

\*/

File directoryName = new File("D:"+File.separator+"test/txt.txt");

System.out.println(directoryName.getPath());

String os = System.getProperty("os.name");

if(os.startsWith("Win")){

System.out.printf("This system is windows");

System.out.println(":::"+os);

}else { //linux

}

/\*\*

\* 2、系统路径的代表符号

\* / linux

\* \ windows

\*/

String fileSeparator = System.getProperty("file.separator");

System.out.println(fileSeparator);

System.out.println(""+File.separator);

}

}