package helper;

import java.io.File;

import java.io.FileInputStream;

import java.io.FilenameFilter;

import java.io.IOException;

import java.io.InputStream;

import java.util.ArrayList;

import java.util.HashMap;

public class FileHelper\_1 {

public static StringBuffer getFileContent(String filepath) {

InputStream is = null;

int i;

char c;

StringBuffer sb = new StringBuffer();

try {

File f = new File(filepath);

System.out.println(f.getCanonicalPath());

if (!f.exists()) {

System.out.println("File Does NOT exist!!");

return sb;

}

is = new FileInputStream(filepath);

byte[] b = new byte[1024];

while ((i = is.read(b)) != -1) {

String s = new String(b);

sb.append(s.trim());

}

} catch (Exception e) {

e.printStackTrace();

} finally {

if (is != null)

try {

is.close();

} catch (IOException e) {

e.printStackTrace();

}

}

return sb;

}

public static HashMap calculateWordWiseCount(StringBuffer content) {

HashMap map = new HashMap();

String[] newlines = content.toString().split("\n");

for (int i = 0; i < newlines.length; i++) {

String[] noun = newlines[i].toString().split(" ");

for (int j = 0; j < noun.length; j++) {

String token=noun[j].replaceAll("\\s+", "").trim();

token=token.replaceAll("^ +| +$| (?= )", "");

if (token.length() > 1) {

if (map.get(token) == null) {

map.put(token, 1);

} else {

int cnt = StringHelper.n2i(map.get(token));

map.put(token, ++cnt);

}

}

}

}

return map;

}

public static ArrayList<String[]> parseFile(String fileName) {

ArrayList<String[]> arr = new ArrayList<String[]>();

StringBuffer sb = getFileContent(fileName);

String[] tokens = sb.toString().split("\\|1234\\|");

for (int i = 0; i < tokens.length; i++) {

String string = tokens[i];

String[] keyTweet = string.split("\\|\\|");

arr.add(keyTweet);

}

return arr;

}

public static ArrayList<String[]> parseFile(String fileName,

String rowDelim, String colDelim) {

ArrayList<String[]> arr = new ArrayList<String[]>();

StringBuffer sb = getFileContent(fileName);

String[] tokens = sb.toString().split(rowDelim);

for (int i = 0; i < tokens.length; i++) {

String string = tokens[i];

String[] keyTweet = string.split(colDelim);

arr.add(keyTweet);

}

return arr;

}

public static File[] getFileList(String dirPath) {

File f = new File(dirPath);

try {

System.out.println("Canonical Path " + f.getCanonicalPath());

} catch (IOException ex) {

ex.printStackTrace();

}

File[] a = f.listFiles();

if (a != null) {

System.out.println(" Got Files " + a.length);

}

return a;

}

// extn=.txt .jpg

public static File[] getFileList(String dirPath, final String extn) {

File f = new File(dirPath);

try {

System.out.println("Canonical Path " + f.getCanonicalPath());

} catch (IOException ex) {

ex.printStackTrace();

}

FilenameFilter textFilter = new FilenameFilter() {

public boolean accept(File dir, String name) {

String lowercaseName = name.toLowerCase();

if (lowercaseName.endsWith(extn)) {

return true;

} else {

return false;

}

}

};

File[] a = f.listFiles(textFilter);

if (a != null) {

System.out.println(" Got Files " + a.length);

}

return a;

}

public static String getSize(File chosenFile) {

long size = chosenFile.length();

return getFileSizeInStr(size);

}

public static String getFileSizeInStr(long size) {

String s = "";

if (size <= 1024) {

s = (size / 1024) + " Kb";

} else {

float mb = (float) (size / 1024);

float mb2 = mb / 1024;

s = Math.round(mb2) + " Mb";

}

return s;

}

}