public class FileResource{

private String myPath;

private String mySource;

private File mySaveFile;

public FileResource(){

initRead();

}

public FileResource(File file){

initRead(file);

}

public FileResource(String filename){

initRead(filename);

}

public FileResource(boolean writable){

if(writable){

initWrite();

}else{

initRead();

}

public FileResource(File file, boolean writable){

if(writable){

initWrite(file);

}

else{

initRead(file);

}

}

public FileResource(String filename, boolean writable){

if(writable) {

initWritable(filename);

}

else {

initRead(filename);

}

}

public Itarable<String>lines() {

return new TextIterable(mySource, "\\n");

}

public Iterable<String>words(){

return new TextIterable(mySource, "\\s+");

public String asString() {

return mySource;

}

public CSVParser getCSVParser(){

return getCSVParser(true);

}

public CSVParser getCSVParser(boolean withHeader){

return getCSVParser(withHeader, ",");

}

public CSVParser getCSVParser(boolean Header, String delimiter){

if (delimiter == null || delimiter.length() != 1) {

throw new ResourceException("FileResource: CSV delimiter must be a single character: " + delimiter);

}

try{

char delim = delimiter.charAt(0);

Reader input = new StringReader(mySource);

if (withHeader) {

return new CSVParser(input, CSVFormat.EXCEL.withHeader().withDelimiter(delim));

}

else{

return new CSVParser(input, CSVFormat.EXCEL.withDelimiter(delim));

}

catch (Exception e) {

throw new ResourceException("FileResource: cannot read " + myPath + " as a CSV file.");

}

}

public Iterable<String>getCSVHeaders(CSVParser parser) {

return parser.getHeaderMap().keySet();

}

public void write(String s) {

ArrayList<String>list = new ArrayList<String>();

list.add(s);

write(list);

}

public void write(StorageResource list){

write(ArrayList<String>(list.data())));

}

public void write(String[]list){

write(new ArrayList<String>(Arrays.asList(list)));

}

public void write(ArrayList<String> list){

if(mySaveFile != null){

//build string to save

StringBuilder sb = new StringBuilder();

for(String s: list){

sb.append(s);

sb.append("\n");

}

//save it locally

mySource += sb.toString();

// save it externally to the file

PrintWriter writer = null;

try{

writer = new PrintWriter(new FileWriter(mySaveFile, true));

writer.println(sb.toString());

}

catch (Exception e){

throw new ResourceException("FileResource: cannot change " + mySaveFile);

}

finally {

try{

if(writer !=null){

writer.close();

}

}

catch(Exception e){

}} } }

private void initRead() {

File f = FileSelector.selectFile();

if(f == null){

throw new ResourceException("FileResource: no file choosen for reading");

} else

{

initRead(f);

}

}

//Create from a given file

private void initRead(File f) {

try{

initRead(f.getCanonicalPath());

}

catch (Exception e){

throw new ResourceException("FileResource: cannot access " + f);

}

}

//Create from the name of a File

private void initRead(String fname) {

try{

myPath = fname;

InputStream is = getClass().getClassLoader().getResourceAsStream(fname);

if(is == null){

is = new FileInputStream(fname);

}

mySource = initFfromStream(is);

}

catch(Exception e){

throw new ResourceException("FileResource: cannot access " + fname);

}

}

//store data(keep in sync with URLResource)

private String initFromStream(InputStream stream) {

BufferedReader buff = null;

try{

buff = new BufferedReader(new InputStreamReader(stream, "UTF-8"));

StringBuilder contents = new StringBuilder();

String line = null;

while ((line = buff.readLine()) !=null){

contents.append(line + "\n");

}

return contents.toString();

}

catch(Exception e){

throw new ResourceException("FileResource: error encountered reading " + myPath, e);

}

finally {

try{

if(buff != null){

buff.close();

}}

catch(Exception e) {

}}}

//prompt user for file for writing

private void initWrite(){

File f = FileSelector.saveFile();

if (f == null){

throw new ResourceException("FileResource: no file choosen for writing");

} else {

initWite(f);

}

}

//create file for writing

private void initWrite(File f) {

try{

mySaveFile = f;

if (f.exists() && f.canWrite()){

initRead(f);

}

else {

mySource = "";

myPath = f.getCanonicalPath();

}

}

catch(Exception e) {

throw new ResourceException("FileResource: cannot access " + f, e);

}

}

// create fle for writing

private void initWrite(String fname){

try{

URL loc = getClass().getClassLoader().getResource(fname);

if(loc != null) {

fname = loc.toString();

}

initWrite(new File(fname));

} catch (Exception e) {

throw new ResourceException("FileResource: cannot access " + fname);

}}}