

File Reading Package

TextReading

All methods are static.

read

Parameter(s): String *path*

Returns: ArrayList<String> *lines*

onlyRead

Parameter(s): int *from*, int *to*, String *path*

Returns: ArrayList<String> *requestedLines*

from and *to* values are **inclusive**.

getLength

Parameter(s): String *path*

Returns: Integer *length*

getSize

Parameter(s): String *path*, int *bytesPerChar*

Returns: Integer *size*

getLineAmount

Parameter(s): String *path*

Returns: Integer *lineAmount*

DataReading

Reads files like .csv

Neutralizer is “ and separator is , as default.

setNeutralizer

Parameter(s): char *neutralizer*

Returns: void

setSeparator

Parameter(s): char *separator*

Returns: void

setDefault

Parameter(s): void

Returns: void

scan

Parameter(s): String *path*

Returns: void

It scans the file and retrieves the data.

getHeader

Parameter(s): void

Returns: ArrayList<String> *headers*

getColumn

Parameter(s): String *header*

Returns: ArrayList<String> *column*

SavfReading

Save file (savf) has this exact pattern for each line:

@parameter=value

scan

Parameter(s): String *path*

Returns: void

getValues

Parameter(s): void

Returns: ArrayList<String> *values*

getParams

Parameter(s): void

Returns: ArrayList<String> *parameters*

getValue

Parameter(s): String *param*

Returns: String *value*

TinfReading

Text Info (tinf) files have this exact pattern:

@title
text
finisher

Finisher is setted as "END-TEXT" as default.

setFinisher

Parameter(s): String *finisher*

Returns: void

setDefault

Parameter(s): void

Returns: void

scan

Parameter(s): String *path*

Returns: void

getTitles

Parameter(s): void

Returns: ArrayList<String> *titles*

getTexts

Parameter(s): void

Returns: ArrayList<String> *texts*

getIndex

Parameter(s): String *title*

Returns: Integer *index*

getRawTexts

Parameter(s): void

Returns: ArrayList<ArrayList<String>>

VitdReading

Visual Improved Text Document (vitd) has this exact pattern:

@header
bg=color,fg=color
text identifier
text
finisher

setFinisher

Parameter(s): String *finisher*

Returns: void

setDefault

Parameter(s): void

Returns: void

scan

Parameter(s): String *path*

Returns: void

getHeaders

Parameter(s): void

Returns: ArrayList<String> *titles*

getTexts

Parameter(s): void

Returns: ArrayList<ArrayList<String>> *texts*

File Writing Package

TextWriting

all methods are static.

write

Parameter(s): String *path*, ArrayList<String> / String[] *lines* / String *line*

Returns: void

append

Parameter(s): String *path*, ArrayList<String> / String[] *lines* / String *line*

Returns: void

appendTo

Parameter(s): String *path*, ArrayList<String> / String[] *lines* / String *line*, int *before*

Returns: void

Adds text inside text which already exists.

change

Parameter(s): String *path*, String *newLine*, int *line*

Returns: void

It changes the requested line.

changeAt

Parameter(s): String *path*, String *changed*, int *line*, int *from*, int *to*

Returns: void

It changes the requested string part with a new one inside the requested line. *from* and *to* values are inclusive.

appendAt

Parameter(s): String *path*, String *appended*, int *line*, int *after*, boolean *leftspace*, boolean *rightspace*

Returns: void

It appends the requested string part inside the requested line after the requested index.

delete

Parameter(s): String *path*, int *line*

Returns: void

It deletes the requested line.

DataWriting

Reads files like .csv

Neutralizer is “ and separator is , as default.

setNeutralizer

Parameter(s): char *neutralizer*

Returns: void

setSeparator

Parameter(s): char *separator*

Returns: void

setDefault

Parameter(s): void

Returns: void

write

Parameter(s): String *path*, ArrayList<ArrayList<String>> *columns*

Returns: void

columns must have the ArrayList<String> of headers in index 0. Then, every single column inside it should be ordered according to headers.

append

Parameter(s): String *path*, ArrayList<String> *lines*

Returns: void

change

Parameter(s): String *path*, String *column*, int *index*, String *newData*

Returns: void

delete

Parameter(s): String *path*, int *index*

Returns: void

addColumn

Parameter(s): String *path*, String *header*, ArrayList<String> *column*

Returns: void

deleteColumn

Parameter(s): String *path*, String *header*

Returns: void

SavfWriting

Save file (savf) has this exact pattern for each line:

@parameter=value

write

Parameter(s): String *path*, ArrayList<String> *params*, ArrayList<String> *values*

Returns: void

change

Parameter(s): String *path*, String *param*, String *newValue*

Returns: void

add

Parameter(s): String *path*, String *param*, String *value*

Returns: void

delete

Parameter(s): String *path*, String *param*

Returns: void

TinfWriting

Text Info (tinf) files have this exact pattern:

@title
text
finisher

Finisher is setted as "END-TEXT" as default.

setFinisher

Parameter(s): String *finisher*

Returns: void

setDefault

Parameter(s): void

Returns: void

write

Parameter(s): String *path*, ArrayList<String> *titles*, ArrayList<String> *texts*

Returns: void

append

Parameter(s): String *path*, ArrayList<String> *titles*, ArrayList<String> *texts*

Returns: void

delete

Parameter(s): String *path*, int *index*

Returns: void

changeTitle

Parameter(s): String *path*, int *index*, String *newTitle*

Returns: void

changeText

Parameter(s): String *path*, int *index*, String *newText*

Returns: void

String Handling Package

ShortedProcesses

console

Shortcut for terminal outputs.

Parameter(s): String *message*

Returns: void

input

Shortcut for terminal inputs.

Parameter(s): void / String *prompt*

Returns: void

PhraseManipulation

where

Parameter(s): String *phrase*, String *requested*

Returns: Integer *location*

howMany

Parameter(s): String *phrase*, String *requested*

Returns: int *amount*

change

Parameter(s): ArrayList<String> *lines*, String *oldPart*, String *newPart*

Returns: ArrayList<String> *editedLines*