Package 'readmoRe'

May 9, 2026

Type Package

Title Utilities for Importing and Manipulating Biomedical Data Files

Version 0.2-15

Date 2025-03-01

Author Vidal Fey [aut, cre]

Maintainer Vidal Fey <vidal.fey@gmail.com>

Description Tools to read various file types into one list of data structures, usually, but not limited to, data frames.
Excel files are read sheet-
wise, i.e., all or a selection of sheets can be read. Field delimiters and decimal separators are determined automatically.

Depends R (>= 3.5.0), R.utils, utils

Imports methods, xml2, readxl, plyr

License GPL-3

Encoding UTF-8

RoxygenNote 7.3.2

NeedsCompilation no

Repository CRAN

Date/Publication 2025-03-01 11:00:02 UTC

Contents

get.nlines — *Determine the number of lines in a (large) text file without importing it.*

Description

Determine the number of lines in a (large) text file without importing it.

Usage

```
get.nlines(file, n = 1, pattern = NULL, incl.header = FALSE)
```

Arguments

file	character of length 1. File path.
n	integer. Maximum number of lines to read when determining the 'skip' value. Passed to 'get.skip'.
pattern	character. Search pattern to find a certain row in 'file'. Passed to 'get.skip'.
incl.header	logical. Should the file header be included in the count? Length of the header will be determined by 'get.skip' and the 'pattern' argument. Defaults to 'FALSE'.

Value

An integer value.

get.sep — *Determine field delimiter in text files*

Description

Determine field delimiter in text files

Usage

```
get.sep(file, n = 1, pattern)
```

Arguments

file	character. Path name of a text file.
n	integer. Number of lines to be read by readLines. Defaults to 1.
pattern	character. Search pattern to find a specific line for determining the delimiter.

Value

If successful, the filed delimiter. If more than on of the possible delimiters is found, an error is returned.

See Also

readLines

`get.skip` *Determine Number of Rows to be Skipped in Text Files*

Description

`get.skip` attempts to determine the number of rows that could be skipped when reading text files.

Usage

```
get.skip(file, n = 1, pattern = NULL)
```

Arguments

`file`	(`character`). The file name.
`n`	(`integer`). The number of lines to be read by `readLines()`.
`pattern`	(`character`). A search pattern like, e.g., a column name that is used to find a particular line in the file to determine the `skip` value.

Value

The `skip` value. If no value is determined 0 (zero) is returned.

See Also

readLines

`read.to.list` *Read various input file formats into a list of data frames. Wrapper function for 'read2list' to automate reading further and avoid errors due to missing folders or files.*

Description

`read.to.list` is meant to act as a universal reading function as it attempts to read a number of different file formats into a list of data frames.

Usage

```
read.to.list(
  dat,
  type,
  folder,
  nsheets = 1,
  sheet = NULL,
  keep.tibble = FALSE,
  skip = 0,
  sep = NULL,
  lines = FALSE,
  dec = NULL,
  ...,
  verbose = TRUE,
  x.verbose = FALSE
)
```

Arguments

`dat`	`character`. File path.
`type`	`character`. File extension to be read: one of ".txt", ".tsv", ".csv", ".vcf", ".gtf", ".gff", ".xls", ".xlsx", ".xdr", ".RData", ".rds", ".rda", ".xml". See details.
`folder`	`character`. Folder where the file is found.
`nsheets`	`integer`. Number of sheets to be read if file is of type ".xls" or ".xlsx". All sheets starting from 1 up to the given number in the respective data file will be read. If more than one file is read this must be be an integer vector with the numbers of sheets in exactly the same order as the files.
`sheet`	`integer` or `list`. Sheet(s) to be read if file is of type ".xls" or ".xlsx". The sheets defined by the given integer in the respective data file will be read. If more than one file is read this must be be a list with the sheet number(s) in exactly the same order as the files. If there are many files and only one sheet vector the same sheet(s) wiöl be read from all files.
`keep.tibble`	`logical`. Should the data from Excel files read with `readxl::read_excel` be coerce to `data.frames` or kept in the original `tibble` format? Defaults to FALSE, i.e., a `data.frame` is returned.

skip	integer. Number of lines to skip from the top of the file.
sep	character. Field delimiter passed to 'read.delim' when reading text files.
lines	lines. Should the file be read line by line into a character vector by readLines()?
dec	character. The decimal separator for numbers.
...	Additional arguments passed to functions.
verbose	logical. Should verbose output be printed?
x.verbose	logical. Should extended verbose output be printed?

Details

Excel files (file extension .xls or .xlsx) will be read by `readxl::read_excel`. A test is attempted to determine whether the input file is genuinely derived from Excel or only named like an nExcel file. If the latter, it will be attempted to read it as text file. Text files are read as tables or by line if `lines` is `TRUE`. For text files, field delimiters and decimal separators are determined automatically if not provided. Files with the extensions .txt", ".tsv", ".csv", ".gtf" and ".gff" are treated and read as text files. VCF files are also treated as text files but can noly be read in full (incl. header) if read by line. Otherwise, if `skip` is `0`, the line with the column names will be determined automatically and the file read as delimited text file. XML files are read by `xml2::read_xml`. ".RData" files are loaded and assigned a name. ".rds" and ".rda" files are read by `readRDS`. ".xdr" files are read by `R.utils::loadObject`.

Value

A list of tibbles/data frames.

See Also

`readLines`

`read.delim`

`read_excel`

`load`

`loadObject`

`readRDS`

`read2list`

Examples

```
# The function readxl::read_excel is used internally to read Excel files.
# The example uses their example data.
readxl_datasets <- readxl::readxl_example("datasets.xlsx")
# A randomly generated data frame was saved to a tab-separated text file
# and two different R object files.
tsv_datasets <- dir(system.file("extdata", package = "readmoRe"), full.names = TRUE)
# All example data are read into a list. From the Excel file, the first
# sheet is read.
dat <- read.to.list(c(readxl_datasets, tsv_datasets))
```

```
# All example data are read into a list. From the Excel file, the first
# 3 sheets are read.
dat <- read.to.list(c(readxl_datasets, tsv_datasets), nsheets=3)
# All example data are read into a list. From the Excel file, sheets 1 and
# 3 are read.
dat <- read.to.list(c(readxl_datasets, tsv_datasets), sheet=c(1, 3))
# From two Excel files, different sheets are read: 1 and 3 from the first
# file and 2 and 3 from the second.
# (For simplicity, the same example file is used.)
dat <- read.to.list(c(readxl_datasets, readxl_datasets), sheet=list(c(1, 3), c(2, 3)))
```

`read2list` *Read various input file formats into a list of data frames*

Description

`read2list` is meant to act as a universal reading function as it attempts to read a number of different file formats into a list of data frames.

Usage

```
read2list(
  dat,
  nsheets = 1,
  sheet = NULL,
  keep.tibble = FALSE,
  skip = 0,
  sep = NULL,
  lines = FALSE,
  dec = NULL,
  ...,
  verbose = TRUE,
  x.verbose = FALSE
)
```

Arguments

`dat`	`character`. File path.
`nsheets`	`integer`. Number of sheets to be read if file is of type ".xls" or ".xlsx". All sheets starting from 1 up to the given number in the respective data file will be read. If more than one file is read this must be be an integer vector with the numbers of sheets in exactly the same order as the files.
`sheet`	`integer` or `list`. Sheet(s) to be read if file is of type ".xls" or ".xlsx". The sheets defined by the given integer in the respective data file will be read. If more than one file is read this must be be a list with the sheet number(s) in exactly the same order as the files. If there are many files and only one sheet vector the same sheet(s) wiöl be read from all files.

keep.tibble	logical. Should the data from Excel files read with readxl::read_excel be coerce to data.frames or kept in the original tibble format? Defaults to FALSE, i.e., a data.frame is returned.
skip	integer. Number of lines to skip from the top of the file.
sep	character. Field delimiter passed to 'read.delim' when reading text files.
lines	lines. Should the file be read line by line into a character vector by readLines()?
dec	character. The decimal separator for numbers.
...	Additional arguments passed to functions.
verbose	logical. Should verbose output be printed?
x.verbose	logical. Should extended verbose output be printed?

See Also

read.delim

read_excel

read.to.list

---

readmoRe *Utilities for data import*

Description

A collection of utilities for reading and importing data into R by performing (usually small) manipulations of data structures such as data frames, matrices and list and automatically determining import parameters.

Details

Package:	readmoRe
Type:	Package
Initial version:	0.1-0
Created:	2011-01-07
License:	GPL-3
LazyLoad:	yes

The main function of the package is read.to.list which reads a number of different file formats into a list of data objects such as data frames, depending on the source file.

Author(s)

Vidal Fey <vidal.fey@gmail.com>

rm.empty.cols *Remove Empty Columns From an Imported Excel Sheet*

Description

rm.empty.cols removes columns that have only NAs *AND* whose names start with a capital 'X' (unless 'na.only' is TRUE in which case all NA columns will be removed).

Usage

```
rm.empty.cols(x, na.only = FALSE)
```

Arguments

x	(data.frame). A data frame resulting from an imported Excel sheet by means of read.xls
na.only	(logical). Should all 'NA' columns be removed and not only those with a column name starting with X as generated by Excel (see details section)?

Details

Empty columns in Excel sheets are imported to NA columns in the resulting data frame. If using gdata::read.xls for reading Excel files, columns that did not have a column name in the spread sheet will result in data frame column names starting with 'X'. rm.empty.cols makes use of these two criteria to identify columns that can safely be removed from the data frame.

Value

A data frame.

rm.newline.chars *Remove 'newline' Characters From Imported Excel Sheets*

Description

rm.newline.chars removes 'newline' characters (\n) from any column of a data frame.

Usage

```
rm.newline.chars(x, verbose = TRUE)
```

Arguments

x	(data.frame). A data frame resulting from an imported Excel sheet by means of read.xls.
verbose	(logical). Should verbose output be printed, defaults to TRUE.

Details

'Newline' characters in data frame rows are read verbatim and will cause rows in output text files to be distributed across two ore more lines. Such characters, entered accidentally or deliberately in the source Excel file, should be avoided. This function removes all 'newline' characters found at the end of a line or replaces them when found within the line text.

Value

A data frame.

Index