Package ‘textreadr’

April 22, 2017

Title Read Text Documents into R

Version 0.5.0

Maintainer Tyler Rinker <tyler.rinker@gmail.com>

Description A small collection of convenience tools for reading text documents into R.

Depends R (>= 3.2.2)

Suggests testthat

Imports antiword, curl, pdftools, readxl, textshape, tools, utils, XML

Date 2017-04-22

License GPL-2

LazyData TRUE

Roxygen list(wrap = FALSE)

RoxygenNote 6.0.1

BugReports https://github.com/trinker/textreadr/issues?state=open

URL https://github.com/trinker/textreadr

Author Tyler Rinker [aut, cre],
Bryan Goodrich [ctb]

RemoteType local

RemoteUrl C:\{ }Users\{ }trinker\{ }GitHub\{ }textreadr

RemoteSha NA

RemoteBranch master

RemoteUsername NA

RemoteRepo NA

R topics documented:

download *Download Documents*

Description

This function enables downloading documents.

Usage

```
download(url, loc = tempdir(), file.out = NULL)
```

Arguments

`url`	The download url(s).
`loc`	Where to put the files.
`file.out`	Option vector of names matching `url`. If this is not given `download` will try to create a name from `url`.

Value

Places a copy of the downloaded document in location specified and returns vector of the locations as string paths.

Examples

```
## Not run:
m <- download(
c('https://cran.r-project.org/web/packages/curl/curl.pdf',
"https://github.com/trinker/textreadr/raw/master/inst/docs/rl10075oralhistoryst002.pdf"),
)

m

## End(Not run)
```

peek *Data Frame Viewing*

Description

peek - Convenience function to view all the columns of the head of a truncated data.frame. peek invisibly returns x. This makes its use ideal in a **dplyr/magrittr** pipeline.

unpeek - Strips out class textreadr so that the entire data.frame will be printed.

Usage

```
peek(x, n = 10, width = 20, ...)

unpeek(x)
```

Arguments

x	A data.frame object.
n	Number of rows to display.
width	The width of the columns to be displayed.
...	For internal use.

Details

By default **dplyr** does not print all columns of a data frame (tbl_df). This makes inspection of data difficult at times, particularly with text string data. peek allows the user to see a truncated head for inspection purposes.

Value

Prints a truncated head but invisibly returns x.

See Also

head

Examples

```
peek(mtcars)
peek(presidential_debates_2012)
```

presidential_debates_2012 *2012 U.S. Presidential Debates*

Description

A dataset containing a cleaned version of all three presidential debates for the 2012 election.

Usage

```
data(presidential_debates_2012)
```

Format

A data frame with 2912 rows and 4 variables

Details

- person. The speaker
- tot. Turn of talk
- dialogue. The words spoken
- time. Variable indicating which of the three debates the dialogue is from

print.textreadr *Prints a textreadr Object*

Description

Prints a textreadr object

Usage

```
## S3 method for class 'textreadr'
print(x, width = 40, ...)
```

Arguments

x	A data.frame textreadr object.
width	The width of the columns to be displayed.
...	Other arguments passed to peek.

read_dir *Read In Multiple Files From a Directory*

Description

Read in multiple files from a directory and create a data.frame.

Usage

```
read_dir(path, pattern = NULL, doc.col = "document", all.files = FALSE,
  recursive = FALSE, ...)
```

Arguments

path	Path to the directory.
pattern	An optional regular expression. Only file names which match the regular expression will be returned.
doc.col	A string naming the document columns (i.e., file names sans file extension).
all.files	Logical. If FALSE, only the names of visible files are returned. If TRUE, all file names will be returned.
recursive	Logical. Should the listing recurse into directories?
...	Other arguments passed to read_document functions.

Value

Returns a data.frame with file names as a document column and content as a text column.

Examples

```
read_dir(system.file("docs/Maas2011/pos", package = "textreadr"))
read_dir(system.file("docs/Maas2011", package = "textreadr"), recursive=TRUE)
```

read_dir_transcript *Read In Multiple Transcript Files From a Directory*

Description

Read in multiple transcript files from a directory and create a data.frame.

Usage

```
read_dir_transcript(path, col.names = c("Document", "Person", "Dialogue"),
  pattern = NULL, all.files = FALSE, recursive = FALSE, skip = 0,
  merge.broke.tot = TRUE, header = FALSE, dash = "", ellipsis = "...",
  quote2bracket = FALSE, rm.empty.rows = TRUE, na = "", sep = NULL,
  comment.char = "", max.person.nchar = 20, ...)
```

Arguments

`path` Path to the directory.

`col.names` A character vector specifying the column names of the transcript columns (document, person, dialogue).

`pattern` An optional regular expression. Only file names which match the regular expression will be returned.

`all.files` Logical. If `FALSE`, only the names of visible files are returned. If `TRUE`, all file names will be returned.

`recursive` Logical. Should the listing recurse into directories?

`skip` Integer; the number of lines of the data file to skip before beginning to read data.

`merge.broke.tot`
logical. If `TRUE` and if the file being read in is .docx with broken space between a single turn of talk read_transcript will attempt to merge these into a single turn of talk.

`header` logical. If `TRUE` the file contains the names of the variables as its first line.

`dash` A character string to replace the en and em dashes special characters (default is to remove).

`ellipsis` A character string to replace the ellipsis special characters.

`quote2bracket` logical. If `TRUE` replaces curly quotes with curly braces (default is `FALSE`). If `FALSE` curly quotes are removed.

`rm.empty.rows` logical. If `TRUE` `read_transcript` attempts to remove empty rows.

`na` A character string to be interpreted as an `NA` value.

`sep` The field separator character. Values on each line of the file are separated by this character. The default of `NULL` instructs `read_transcript` to use a separator suitable for the file type being read in.

`comment.char` A character vector of length one containing a single character or an empty string. Use `""` to turn off the interpretation of comments altogether.

`max.person.nchar`
The max number of characters long names are expected to be. This information is used to warn the user if a separat appears beyond this length in the text.

`...` ignored.

Value

Returns a dataframe of documents, dialogue, and people.

See Also

read_transcript

Examples

```
skips <- c(0, 1, 1, 0, 0, 1)
path <- system.file("docs/transcripts", package = 'textreadr')
textreadr::peek(read_dir_transcript(path, skip = skips), Inf)

## Not run:
## with additional  cleaning
```

```
library(tidyverse, textshape, textclean)

path %>%
    read_dir_transcript(skip = skips) %>%
    textclean::filter_row("Person", "^\\[") %>%
    mutate(
        Person = stringi::stri_replace_all_regex(Person, "(^/\\s*)|(:\\s*$)", "") %>%
            trimws(),
        Dialogue = stringi::stri_replace_all_regex(Dialogue, "(^/\\s*)", "")
    ) %>%
    peek(Inf)

## End(Not run)
```

read_doc *Read in .doc Content*

Description

Read in the content from a .doc file using antiword via the **antiword** package.

Usage

```
read_doc(file, skip = 0, remove.empty = TRUE, format = TRUE, ...)
```

Arguments

`file`	The path to the .doc file.
`skip`	The number of lines to skip.
`remove.empty`	logical. If `TRUE` empty elements in the vector are returned.
`format`	logical. If `TRUE` the output will keep doc formatting (e.g., bold, italics, underlined). This corresponds to the `-f` flag in antiword.
`...`	ignored.

Value

Returns a character vector.

Examples

```
## Not run:
x <- system.file("docs/Yasmine_Interview_Transcript.doc",
    package = "textreadr")
read_doc(x)

## End(Not run)
```

read_document *Generic Function to Read in a Document*

Description

Generic function to read in a .pdf, .txt, .docx, or .doc file.

Usage

```
read_document(file, combine = FALSE, ...)
```

Arguments

file	The path to the a .pdf, .txt, .docx, or .doc file.
combine	logical. If TRUE the vector is concatenated into a single string via combine
...	Other arguments passed to read_pdf, read_docx, read_doc, or readLines.

Value

Returns a list of string vectors.

Examples

```
## .pdf
pdf_doc <- system.file("docs/rl10075oralhistoryst002.pdf",
    package = "textreadr")
read_document(pdf_doc)

## .docx
docx_doc <- system.file("docs/Yasmine_Interview_Transcript.docx",
    package = "textreadr")
read_document(docx_doc)

## .txt
txt_doc <- system.file('docs/textreadr_creed.txt', package = "textreadr")
read_document(txt_doc)

## Not run:
doc_doc <- system.file("docs/Yasmine_Interview_Transcript.doc",
    package = "textreadr")
read_document(doc_doc)

## End(Not run)
```

read_docx *Read in .docx Content*

Description

Read in the content from a .docx file.

Usage

```
read_docx(file, skip = 0)
```

Arguments

file	The path to the .docx file.
skip	The number of lines to skip.

Value

Returns a character vector.

Author(s)

Bryan Goodrich and Tyler Rinker <tyler.rinker@gmail.com>.

Examples

```
## Not run:
x <- "http://www.cybersmart.gov.au/~/media/9999BCDEA99F40DD8170AAD978C8D2F9.docx"
out <- download(x)
(txt <- read_docx(out))

## End(Not run)
```

read_pdf *Read a Portable Document Format into R*

Description

A wrapper for pdf_text to read PDFs into R.

Usage

```
read_pdf(file, skip = 0)
```

Arguments

file	A path to a PDF file.
skip	Integer; the number of lines of the data file to skip before beginning to read data.

Value

Returns a `data.frame` with the page number (`page_id`), line number (`element_id`), and the `text`.

Note

A word of caution from Carl Witthoft" "Just a warning to others who may be hoping to extract data: PDF is a container, not a format. If the original document does not contain actual text, as opposed to bitmapped images of text or possibly even uglier things than I can imagine, nothing other than OCR can help you."

See Also

`readPDF`

Examples

```
pdf_dat <- read_pdf(
    system.file("docs/rl10075oralhistoryst002.pdf", package = "textreadr")
)

pdf_dat_b <- read_pdf(
    system.file("docs/rl10075oralhistoryst002.pdf", package = "textreadr"),
    skip = 1
)

## Not run:
library(textshape)
system.file("docs/rl10075oralhistoryst002.pdf", package = "textreadr") %>%
    read_pdf(1) %>%
    `[[`('text') %>%
    head(-1) %>%
    textshape::combine() %>%
    gsub("([A-Z])( )([A-Z])", "\\1_\\3", .) %>%
    strsplit("(-| )(?=[A-Z_]+:)", perl=TRUE) %>%
    `[[`(1) %>%
    textshape::split_transcript()

## End(Not run)
```

`read_transcript` *Read Transcripts Into R*

Description

Read .docx, .csv, .xlsx, .xlsx, or .txt transcript style files into R.

Usage

```
read_transcript(file, col.names = c("Person", "Dialogue"), text.var = NULL,
  merge.broke.tot = TRUE, header = FALSE, dash = "", ellipsis = "...",
  quote2bracket = FALSE, rm.empty.rows = TRUE, na = "", sep = NULL,
  skip = 0, text, comment.char = "", max.person.nchar = 20, ...)
```

Arguments

`file` The name of the file which the data are to be read from. Each row of the table appears as one line of the file. If it does not contain an absolute path, the file name is relative to the current working directory, `getwd()`.

`col.names` A character vector specifying the column names of the transcript columns.

`text.var` A character string specifying the name of the text variable will ensure that variable is classed as character. If `NULL` `read_transcript` attempts to guess the text.variable (dialogue).

`merge.broke.tot` logical. If `TRUE` and if the file being read in is .docx with broken space between a single turn of talk read_transcript will attempt to merge these into a single turn of talk.

`header` logical. If `TRUE` the file contains the names of the variables as its first line.

`dash` A character string to replace the en and em dashes special characters (default is to remove).

`ellipsis` A character string to replace the ellipsis special characters.

`quote2bracket` logical. If `TRUE` replaces curly quotes with curly braces (default is `FALSE`). If `FALSE` curly quotes are removed.

`rm.empty.rows` logical. If `TRUE` `read_transcript` attempts to remove empty rows.

`na` A character string to be interpreted as an `NA` value.

`sep` The field separator character. Values on each line of the file are separated by this character. The default of `NULL` instructs `read_transcript` to use a separator suitable for the file type being read in.

`skip` Integer; the number of lines of the data file to skip before beginning to read data.

`text` Character string: if file is not supplied and this is, then data are read from the value of text. Notice that a literal string can be used to include (small) data sets within R code.

`comment.char` A character vector of length one containing a single character or an empty string. Use `""` to turn off the interpretation of comments altogether.

`max.person.nchar` The max number of characters long names are expected to be. This information is used to warn the user if a separat appears beyond this length in the text.

`...` Further arguments to be passed to `read.table`, `read_excel`, or `read_doc`.

Value

Returns a dataframe of dialogue and people.

Warning

`read_transcript` may contain errors if the file being read in is .docx. The researcher should carefully investigate each transcript for errors before further parsing the data.

Note

If a transcript is a .docx file read_transcript expects two columns (generally person and dialogue) with some sort of separator (default is colon separator). .doc files must be converted to .docx before reading in.

Author(s)

Bryan Goodrich and Tyler Rinker <tyler.rinker@gmail.com>.

References

https://github.com/trinker/qdap/wiki/Reading-.docx-%5BMS-Word%5D-Transcripts-into-R

Examples

```
(doc1 <- system.file("docs/trans1.docx", package = "textreadr"))
(doc2 <- system.file("docs/trans2.docx", package = "textreadr"))
(doc3 <- system.file("docs/trans3.docx", package = "textreadr"))
(doc4 <- system.file("docs/trans4.xlsx", package = "textreadr"))
(doc5 <- system.file("docs/trans5.xls", package = "textreadr"))
(doc6 <- system.file("docs/trans6.doc", package = "textreadr"))

dat1 <- read_transcript(doc1)
dat2 <- read_transcript(doc1, col.names = c("person", "dialogue"))

## read_transcript(doc2) #throws an error (need skip)
dat3 <- read_transcript(doc2, skip = 1)

## read_transcript(doc3, skip = 1) #incorrect read; wrong sep
dat4 <- read_transcript(doc3, sep = "-", skip = 1)

## xlsx/xls format
dat5 <- read_transcript(doc4)
dat6 <- read_transcript(doc5)

## Not run:
## MS doc format (must have antiword installed)
dat7 <- read_transcript(doc6) ## need to skip Researcher
dat8 <- read_transcript(doc6, skip = 1)

## End(Not run)

trans <- "sam: Computer is fun. Not too fun.
greg: No it's not, it's dumb.
teacher: What should we do?
sam: You liar, it stinks!"

read_transcript(text=trans)

## Read in text specify spaces as sep
## EXAMPLE 1
read_transcript(text="34    The New York Times reports a lot of words here.
12    Greenwire reports a lot of words.
31    Only three words.
 2    The Financial Times reports a lot of words.
 9    Greenwire short.
13    The New York Times reports a lot of words again.",
    col.names = c("NO", "ARTICLE"), sep = "   ")

## EXAMPLE 2
read_transcript(text="34..    The New York Times reports a lot of words here.
12..    Greenwire reports a lot of words.
31..    Only three words.
 2..    The Financial Times reports a lot of words.
 9..    Greenwire short.
13..    The New York Times reports a lot of words again.",
    col.names = c("NO", "ARTICLE"), sep = "\\.\\.")

## Real Example
real_dat <- read_transcript(
    system.file("docs/Yasmine_Interview_Transcript.docx", package = "textreadr"),
    skip = 19
)
```

`textreadr` *Read Text Documents into R*

Description

A small collection of convenience tools for reading text documents into R.

Index