Package 'diffrprojects'

November 1, 2016

Title Using diffr for more than two files

Date 2016-10-02

Version 0.1.11.90000

Description This is a description still to be done but to prevent checks about complaining about to short descriptions this does not simply read TBD.

Depends R (>= 3.0.0), stringb (>= 0.1.11), rtext (>= 0.1.16)

License MIT + file LICENSE

LazyData TRUE

Imports R6 (>= 2.1.2), hellno (>= 0.0.1), dplyr(>= 0.5.0), data.table (>= 1.9.6), dtplyr (>= 0.0.1), Rcpp (>= 0.12.6), stringdist (>= 0.9.4.1), tidyr(>= 0.6.0), RSQLite (>= 1.0.0), magrittr, stats, graphics

Suggests testthat, knitr, rmarkdown

BugReports https://github.com/petermeissner/diffrprojects/issues

URL https://github.com/petermeissner/diffrprojects

RoxygenNote 5.0.1

VignetteBuilder knitr

LinkingTo Rcpp

Author Peter Meissner [aut, cre], Ulrich Sieberer [cph], University of Konstanz [cph]

Maintainer Peter Meissner <retep.meissner@gmail.com>

R topics documented:

`as.data.frame.alignment_data_list` *as.data.frame method for for named lists of data.frames*

Description

as.data.frame method for for named lists of data.frames

Usage

```
## S3 method for class 'alignment_data_list'
as.data.frame(x, row.names = NULL,
  optional = FALSE, ...)
```

Arguments

`x`	any R object.
`row.names`	`NULL` or a character vector giving the row names for the data frame. Missing values are not allowed.
`optional`	logical. If `TRUE`, setting row names and converting column names (to syntactic names: see `make.names`) is optional. Note that all of R's **base** package `as.data.frame()` methods use `optional` only for column names treatment, basically with the meaning of `data.frame(*, check.names = !optional)`.
`...`	additional arguments to be passed to or from methods.

as.data.frame.alignment_list

as.data.frame method for for named lists of data.frames

Description

as.data.frame method for for named lists of data.frames

Usage

```
## S3 method for class 'alignment_list'
as.data.frame(x, row.names = NULL,
  optional = FALSE, ...)
```

Arguments

x	any R object.
row.names	NULL or a character vector giving the row names for the data frame. Missing values are not allowed.
optional	logical. If TRUE, setting row names and converting column names (to syntactic names: see make.names) is optional. Note that all of R's **base** package as.data.frame() methods use optional only for column names treatment, basically with the meaning of data.frame(*, check.names = !optional).
...	additional arguments to be passed to or from methods.

as.data.frame.named_df_list

as.data.frame method for for named lists of data.frames

Description

as.data.frame method for for named lists of data.frames

Usage

```
## S3 method for class 'named_df_list'
as.data.frame(x, row.names = NULL, optional = FALSE,
  dfnamevar = "name", ...)
```

Arguments

x	any R object.
row.names	NULL or a character vector giving the row names for the data frame. Missing values are not allowed.
optional	logical. If TRUE, setting row names and converting column names (to syntactic names: see make.names) is optional. Note that all of R's **base** package as.data.frame() methods use optional only for column names treatment, basically with the meaning of data.frame(*, check.names = !optional).
dfnamevar	in which variable should list item names be saved
...	additional arguments to be passed to or from methods.

`choose_options` *(choose from a number of pre-sorted options) takes a vector pair of toki1 / toki2 and a vector pair of res_token_i_1 / res_token_i_2 and chooses so that each 1st and exh 2nd value only is used where res_token_i_x identiefies already used items.*

Description

(choose from a number of pre-sorted options) takes a vector pair of toki1 / toki2 and a vector pair of res_token_i_1 / res_token_i_2 and chooses so that each 1st and exh 2nd value only is used where res_token_i_x identiefies already used items.

Usage

```
choose_options(toki1, toki2, res_token_i_1, res_token_i_2)
```

Arguments

toki1	first number of number pair to choose from
toki2	second number of number pair to choose from
res_token_i_1	already used first numbers
res_token_i_2	already used second numbers // @keywords internal

`diffrproject` *class for diffrproject*

Description

class for diffrproject

class for dp_align

class for dp_base

class for dp_inherit

class for dp_base

Usage

```
diffrproject

dp_align

dp_base

dp_inherit

dp_loadsave
```

Format

`R6Class` creator object.

Value

Object of `diffrproject`

Object of `dp_align`

Object of `dp_base`

Object of `dp_align`

Object of `dp_loadsave`

The diffrprojects class family

Diffrproject consists of an set of R6 classes that are conencted by inheritance. Each class handles a different set of functionalities that are modular.

R6_rtext_extended A class that has nothing to do per se with diffrprojects. It merely adds some basic features to the base R6 class (debugging, hashing, getting fields and handling warnings and messages as well as listing content). This class is imported from rtext package

dp_base [inherits from rtext::R6_rtext_extended] This class forms the foundation of all diffrpojects (dp_xxx) classes by implementing data fields for meta data, texts, data on texts, links between texts, alignment of text tokens, and data on the alignment of text tokens. Furthermore it implements methods add, delete, code, and link texts or to aggregate text data on text token level.

dp_loadsave [inherits from dp_base] This class allows for loading and saving diffrprojects from and to Rdata files.

dp_export [inherits from dp_loadsave] This class provides methods for exporting and importing to and from RSQLite.

dp_align [inherits from dp_export] This is one of the workhorses of diffrprojects. The methods of this class allow for adding, deleting or computing alignments between text tokens (e.g. words or lines or sentences or characters or paragraphs, or some other way to split text into chunks). Furthermore it allows to also assign data to individual alignments (a connection beween two token of text from different text versions).

dp_inherit [inherits from dp_align] The text_data_inherit method added by this class allows to copy text data from one token of a text version to another token of another text version channeled through aligments with zero distance. Conflicting codings (a text might have multiple codings stemming from several links and from direct coding of the text) are resolved by the fact that text codings are accompanied by a hierarchy level that defaults to zero and gets decreased by one every time the coding is inherited by a token.

diffrproject [inherits from dp_inherit] Just a wrapper inheriting from dp_inherit to have a less technical name at the end of the inheritance chain.

`diff_align` *algining texts*

Description

Function aligns two texts side by side as a data.frame with change type and distance given as well

Usage

```
diff_align(text1 = NULL, text2 = NULL, tokenizer = NULL, ignore = NULL,
  clean = NULL, distance = c("lv", "osa", "dl", "hamming", "lcs", "qgram",
  "cosine", "jaccard", "jw", "soundex"), useBytes = FALSE, weight = c(d = 1,
  i = 1, s = 1, t = 1), maxDist = 0, q = 1, p = 0,
  nthread = getOption("sd_num_thread"), verbose = TRUE, ...)
```

Arguments

`text1`	first text
`text2`	second text
`tokenizer`	defaults to NULL which will trigger linewise tokenization; accepts a function that turns a text into a token data frame; a token data frame has at least three columns: from (first character of token), to (last character of token) token (the token)

ignore	defaults to NULL which means that nothing is ignored; function that accepts a token data frame (see above) and returns a possibly subseted data frame of hte same form
clean	defaults to NULL which means that nothing cleaned; accepts a function that takes a vector of tokens and returns a vector of same length - potentially clean up
distance	defaults to Levenshtein ("lv"); see amatch, stringdist-metrics, stringdist
useBytes	Perform byte-wise comparison, see stringdist-encoding.
weight	For method='osa' or 'dl', the penalty for deletion, insertion, substitution and transposition, in that order. When method='lv', the penalty for transposition is ignored. When method='jw', the weights associated with characters of a, characters from b and the transposition weight, in that order. Weights must be positive and not exceed 1. weight is ignored completely when method='hamming', 'qgram', 'cosine', 'Jaccard', 'lcs', or soundex.
maxDist	[DEPRECATED AND WILL BE REMOVED|2016] Currently kept for backward compatibility. It does not offer any speed gain. (In fact, it currently slows things down when set to anything different from Inf).
q	Size of the q-gram; must be nonnegative. Only applies to method='qgram', 'jaccard' or 'cosine'.
p	Penalty factor for Jaro-Winkler distance. The valid range for p is 0 <= p <= 0.25. If p=0 (default), the Jaro-distance is returned. Applies only to method='jw'.
nthread	Maximum number of threads to use. By default, a sensible number of threads is chosen, see stringdist-parallelization.
verbose	should function report on its doings via messages or not
...	further arguments passed through to distance function

Value

dataframe with tokens aligned according to distance

---

dp_export *R6 class - linking text and data*

Description

R6 class - linking text and data

Usage

```
dp_export
```

Format

R6Class object.

Value

Object of R6Class

`dp_text_base_data` *function providing basic information on texts within diffrproject*

Description

function providing basic information on texts within diffrproject

Usage

```
dp_text_base_data(dp)
```

Arguments

dp	a diffrproject object

`dummyimport` *imports*

Description

imports

Usage

```
dummyimport()
```

`get_private` *accessing private from R6 object*

Description

accessing private from R6 object

Usage

```
get_private(x)
```

Arguments

x	R6 object to access private from

Source

http://stackoverflow.com/a/38578080/1144966

push_text_char_data *push char_data of one rtext objet to another*

Description

Function that takes a rtext object pulls specific char_data from it and pushes this information to another rtext object.

Usage

```
push_text_char_data(from_text = NULL, to_text = NULL, from_token = NULL,
  to_token = NULL, from_i = NULL, to_i = NULL, x = NULL, warn = TRUE)
```

Arguments

from_text	text to pull data from
to_text	text to push data to
from_token	token of text to pull data from (e.g.: data.frame(from=1, to=4))
to_token	token of text to push data to (e.g.: data.frame(from=1, to=4))
from_i	index of characters to pull data from
to_i	index of characters to push data to
x	name of the char_data variable to pull and push - defaults to NULL which will result in cycling through all availible variables
warn	should function warn about non-uniform pull values (those will not be pushed to the other text)

Details

Note, that this is an intelligent function.

It will e.g. always decrease the hierarchy level (hl) found when pulling and decrease it before pushing it forward therewith allowing that already present coding might take priority over those pushed.

Furthermore, the function will only push values if the pulled values are all the same. Since, character index lengths that are used for pulling and pushing might differ in length there is no straight forward rule to translate non uniform value sequences in value sequnces of differing length. Note, that of cause the values might differ between char_data variables but not within. In case of non-uniformity the function will simply do nothing.

sort_alignment *function sorting alignment data according to token index*

Description

function sorting alignment data according to token index

Usage

```
sort_alignment(x, ti1 = NULL, ti2 = NULL, first = TRUE)
```

Arguments

x	data.frame to be sorted
ti1	either NULL (default): first column of x is used as first token index for sorting; a character vector specifying the column to be used as first token index; or a numeric vector of length nrow(x) to be use as first token index
ti2	either NULL (default): second column of x is used as second token index for sorting; a character vector specifying the column to be used as second token index; or a numeric vector of length nrow(x) to be use as second token index
first	should first text or second text be given priority

write_numerous_parts_to_table *function writing numerous parts of table to database*

Description

function writing numerous parts of table to database

Usage

```
write_numerous_parts_to_table(x, con, table_name, meta = data.frame())
```

Arguments

x	parts to be written
con	connection to database
table_name	of the table
meta	additional information to be attachesd to table parts

Index