Package ‘rtext’

January 28, 2021

Title R6 Objects for Text and Data
Date 2021-01-27
Version 0.1.22

Description For natural language processing and analysis of qualitative text coding structures which provide a way to bind together text and text data are fundamental. The package provides such a structure and accompanying methods in form of R6 objects. The 'rtext' class allows for text handling and text coding (character or regex based) including data updates on text transformations as well as aggregation on various levels. Furthermore, the usage of R6 enables inheritance and passing by reference which should enable 'rtext' instances to be used as back-end for R based graphical text editors or text coding GUIs.

Depends R (>= 3.0.0), stringb (>= 0.1.13)
License MIT + file LICENSE
LazyData TRUE

Imports R6 (>= 2.1.2), hellno (>= 0.0.1), magrittr (>= 1.5), Rcpp (>= 0.12.5), digest (>= 0.6.9), RSQLite (>= 1.0.0), stats, graphics
Suggests testthat, knitr, rmarkdown

BugReports https://github.com/petermeissner/rtext/issues
URL https://github.com/petermeissner/rtext
RoxygenNote 7.1.1
LinkingTo Rcpp

NeedsCompilation yes

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

Maintainer Peter Meissner <retep.meissner@gmail.com>
Repository CRAN
Date/Publication 2021-01-28 07:40:02 UTC
R topics documented:

modus .......................................................... 2
plot.rtext ......................................................... 2
prometheus_early .............................................. 3
prometheus_late ............................................... 3
R6_rtext_extended ............................................ 4
rtext ............................................................. 5
rtext_base ....................................................... 7
rtext_export ................................................... 10
rtext_loadsave ............................................... 11
rtext_tokenize ............................................... 12
text_tokenize.rtext ........................................... 14

Index 15

modus  function giving back the mode

Description

function giving back the mode

Usage

modus(x, multimodal = FALSE, warn = TRUE)

Arguments

x vector to get mode for
multimodal wether or not all modes should be returned in case of more than one
warn should the function warn about multimodal outcomes?

plot.rtext  function for plotting rtext

Description

function for plotting rtext

Usage

## S3 method for class 'rtext'
plot(x, y = NULL, lines = TRUE, col = "#ED4C4CA0", add = FALSE, ...)
**Arguments**

- **x**: object of class rtext
- **y**: char_data to be plotted
- **lines**: vector of integer listing the lines to be plotted
- **col**: color of the char_data variable to be highlighted
- **add**: add data to an already existing plot?
- ... further parameters passed through to initial plot

---

**Description**

prometheus early version

**Usage**

`prometheus_early`

**Format**

An object of class character of length 1.

**Source**


---

**Description**

prometheus late version

**Usage**

`prometheus_late`

**Format**

An object of class character of length 1.

**Source**

**R6_rtext_extended**  
*extended R6 class*

---

**Description**
extended R6 class

**Format**

*R6Class* object.

**Value**
Object of *R6Class*

**Methods**

**Public methods:**

- `R6_rtext_extended$get()`  
- `R6_rtext_extended$debug()`  
- `R6_rtext_extended$ls()`  
- `R6_rtext_extended$message()`  
- `R6_rtext_extended$warning()`  
- `R6_rtext_extended$clone()`

**Method get():**

*Usage:*

`R6_rtext_extended$get(name = NULL)`

**Method debug():**

*Usage:*

`R6_rtext_extended$debug(pos = 1)`

**Method ls():**

*Usage:*

`R6_rtext_extended$ls(what = c("self", "private"), class = NULL)`

**Method message():**

*Usage:*

`R6_rtext_extended$message(x, ...)`

**Method warning():**

*Usage:*

`R6_rtext_extended$warning(x, ...)`
Method `clone()`: The objects of this class are cloneable with this method.

Usage:

```
R6_rtext_extended$clone(deep = FALSE)
```

Arguments:

- `deep` Whether to make a deep clone.

See Also

`rtext`

---

rtext  

**R6 class - linking text and data**

---

Description

R6 class - linking text and data

R6 class - linking text and data

Format

An `R6Class` generator object.

Value

Object of `R6Class`

The rtext class family

Rtext consists of an set of R6 classes that are connected by inheritance. Each class handles a different set of functionalities that are - despite needing the data structure provided by `rtext_base` - independent.

- **R6_rtext_extended** A class that has nothing to do per se with rtext but merely adds some basic features to the base R6 class (debugging, hashing, getting fields and handling warnings and messages as well as listing content)

- **rtext_base** [inherits from R6_rtext_extended] The foundation of the rtext class. This class allows to load and store text, its meta data, as well as data about the text in a character by character level.

- **rtext_loadsave** [inherits from rtext_base] Adds load and save methods for loading and saving rtext objects (text and data) into/from Rdata files.

- **rtext_export** [inherits from rtext_loadsave] Adds methods to import and export from and to SQLite databases - like load and save but for SQLite.

- **rtext_tokenize** [inherits from rtext_export] Adds methods to aggregate character level data onto token level. (the text itself can be tokenized via S3 methods from the stringb package - e.g. `text_tokenize_words()`)

- **rtext** [inherits from rtext_tokenize] Adds no new features at all but is just a handy label sitting on top of all the functionality provided by the inheritance chain.
Super classes

```r
rtext::R6_rtext_extended -> rtext::rtext_base -> rtext::rtext_loadsave -> rtext::rtext_export
-> rtext::rtext_tokenize -> rtext
```

Methods

Public methods:

- `rtext$clone()`

**Method clone():** The objects of this class are cloneable with this method.

**Usage:**

```r
rtext$clone(deep = FALSE)
```

**Arguments:**

- `deep` Whether to make a deep clone.

Examples

```r
# initialize (with text or file)
quote_text <-
"Outside of a dog, a book is man's best friend. Inside of a dog it's too dark to read."
quote <- rtext$new(text = quote_text)

# add some data
quote$char_data_set("first", 1, TRUE)
quote$char_data_set("last", quote$char_length(), TRUE)

# get the data
quote$char_data_get()

# transform text
quote$char_add("[this is an insertion] \n", 47)

# get the data again (see, the data moved along with the text)
quote$text_get()
quote$char_data_get()

# do some convenience coding (via regular expressions)
quote$char_data_set_regex("dog_friend", "dog", "dog")
quote$char_data_set_regex("dog_friend", "friend", "friend")
quote$char_data_get()

# aggregate data by regex pattern
quote$tokenize_data_regex(split="(dog)|(friend)", non_token = TRUE, join = "full")

# aggregate data by words
quote$tokenize_data_words(non_token = TRUE, join="full")

# aggregate data by lines
quote$tokenize_data_lines()
```
# plotting and data highlighting
plot(quote, "dog_friend")

# adding further data to the plot
plot(quote, "dog_friend")
plot(quote, "first", col="steelblue", add=TRUE)
plot(quote, "last", col="steelblue", add=TRUE)

---

**rtext_base**

**rtext_base : basic workhorse for rtext**

### Description

\texttt{rtext\_base} : basic workhorse for \texttt{rtext}

### Format

\texttt{R6Class} object.

### Value

Object of \texttt{R6Class}

### Super class

\texttt{rtext::R6\_rtext\_extended} \rightarrow \texttt{rtext\_base}

### Methods

**Public methods:**

- \texttt{rtext\_base$new()}
- \texttt{rtext\_base$info()}
- \texttt{rtext\_base$text\_show()}
- \texttt{rtext\_base$text\_get()}
- \texttt{rtext\_base$text\_get\_lines()}
- \texttt{rtext\_base$char\_get()}
- \texttt{rtext\_base$char\_add()}
- \texttt{rtext\_base$char\_delete()}
- \texttt{rtext\_base$char\_replace()}
- \texttt{rtext\_base$char\_length()}
- \texttt{rtext\_base$char\_data\_set()}
- \texttt{rtext\_base$char\_data\_set\_regex()}
- \texttt{rtext\_base$char\_data\_get()}
• `rtext_base$hash_get()`
• `rtext_base$clone()`

**Method new():**

*Usage:*

```r
rtext_base$new(
  text = NULL,
  text_file = NULL,
  encoding = "UTF-8",
  id = NULL,
  save_file = NULL,
  verbose = TRUE
)
```

**Method info():**

*Usage:*

```r
rtext_base$info()
```

**Method text_show():**

*Usage:*

```r
rtext_base$text_show(
  length = 500,
  from = NULL,
  to = NULL,
  coll = FALSE,
  wrap = FALSE
)
```

**Method text_get():**

*Usage:*

```r
rtext_base$text_get(length = Inf, from = NULL, to = NULL, split = NULL)
```

**Method text_get_lines():**

*Usage:*

```r
rtext_base$text_get_lines(length = Inf, from = NULL, to = NULL)
```

**Method char_get():**

*Usage:*

```r
rtext_base$char_get(length = Inf, from = NULL, to = NULL, raw = FALSE)
```

**Method char_add():**

*Usage:*

```r
rtext_base$char_add(what = NULL, after = NULL)
```

**Method char_delete():**

*Usage:*

```r
```
Method char_delete():
Usage:
rtext_base$char_delete(n = NULL, from = NULL, to = NULL)

Method char_replace():
Usage:
rtext_base$char_replace(from = NULL, to = NULL, by = NULL)

Method char_length():
Usage:
rtext_base$char_length()

Method char_data_set():
Usage:
rtext_base$char_data_set(x = NULL, i = NULL, val = NA, hl = 0)

Method char_data_set_regex():
Usage:
rtext_base$char_data_set_regex(x = NULL, pattern = NULL, val = NA, hl = 0, ...)

Method char_data_get():
Usage:
rtext_base$char_data_get(from = 1, to = Inf, x = NULL, full = FALSE)

Method hash_get():
Usage:
rtext_base$hash_get(name = "")

Method clone(): The objects of this class are cloneable with this method.
Usage:
rtext_base$clone(deep = FALSE)

Arguments:
deepl Whether to make a deep clone.

See Also
rtext
rtext_export

R6 class - linking text and data

Description
R6 class - linking text and data
R6 class - linking text and data

Format
R6Class object.

Value
Object of R6Class

Super classes
rtext::R6_rtext_extended -> rtext::rtext_base -> rtext::rtext_loadsave -> rtext_export

Methods
Public methods:
• rtext_export$export_csv()
• rtext_export$import_csv()
• rtext_export$export_sqlite()
• rtext_export$import_sqlite()
• rtext_export$clone()

Method export_csv():
Usage:
rtext_export$export_csv(folder_name = "")

Method import_csv():
Usage:
rtext_export$import_csv(folder_name = "")

Method export_sqlite():
Usage:
rtext_export$export_sqlite(db_name = "")

Method import_sqlite():
Usage:
rtext_export$import_sqlite(db_name = ")"
Method clone(): The objects of this class are cloneable with this method.

Usage:
```
rtexlexport$clone(deep = FALSE)
```

Arguments:
- deep: Whether to make a deep clone.

See Also
- rtext

---

**Description**

R6 class - load and save methods for rtext

**Format**

R6Class object.

**Value**

Object of R6Class

**Super classes**

rtext::R6_rtext_extended -> rtext::rtext_base -> rtext_loadsave

**Methods**

Public methods:
- rtext_loadsave$save()
- rtext_loadsave$load()
- rtext_loadsave$clone()

Method save():

Usage:
```
rtexlexport$save(file = NULL, id = NULL)
```

Method load():

Usage:
```
rtexlexport$load(file = NULL)
```

Method clone(): The objects of this class are cloneable with this method.
Usage:
`rtext_loadsave$clone(deep = FALSE)`

Arguments:
depth Whether to make a deep clone.

See Also
- `rtext`

---

**rtext_tokenize**  
*R6 class - linking text and data*

**Description**
- R6 class - linking text and data
- R6 class - linking text and data

**Format**
- `R6Class` object.

**Value**
- Object of `R6Class`

**Super classes**
- `rtext::R6_rtext_extended` -> `rtext::rtext_base` -> `rtext::rtext_loadsave` -> `rtext::rtext_export` -> `rtext_tokenize`

**Methods**

**Public methods:**
- `rtext_tokenize$tokenize_data_regex()`
- `rtext_tokenize$tokenize_data_sequences()`
- `rtext_tokenize$tokenize_data_words()`
- `rtext_tokenize$tokenize_data_lines()`
- `rtext_tokenize$clone()`

**Method** `tokenize_data_regex()`:

Usage:
Method `tokenize_data_sequences()`:

Usage:
```
rtex_text_tokenize$tokenize_data_sequences(
  token,
  join = c("full", "left", "right", ""),
  aggregate_function = NULL,
  ...
)
```

Method `tokenize_data_words()`:

Usage:
```
rtex_text_tokenize$tokenize_data_words(
  split = \W+,
  ignore.case = FALSE,
  fixed = FALSE,
  perl = FALSE,
  useBytes = FALSE,
  non_token = FALSE,
  join = c("full", "left", "right", ""),
  aggregate_function = NULL,
  ...
)
```

Method `tokenize_data_lines()`:

Usage:
```
rtex_text_tokenize$tokenize_data_lines(
  split = \n,
  ignore.case = FALSE,
  fixed = FALSE,
  perl = FALSE,
  useBytes = FALSE,
  non_token = FALSE,
  join = c("full", "left", "right", ""),
  aggregate_function = NULL,
  ...
)
```
Method clone(): The objects of this class are cloneable with this method.

Usage:
rttext_tokenize$clone(deep = FALSE)

Arguments:
deeper Whether to make a deep clone.

See Also
rttext

text_tokenize.rtext function tokenizing rtext objects

Description
function tokenizing rtext objects

Usage
## S3 method for class 'rtext'
text_tokenize(
  string,
  regex = NULL,
  ignore.case = FALSE,
  fixed = FALSE,
  perl = FALSE,
  useBytes = FALSE,
  non_token = FALSE
)

Arguments
string text to be tokenized
regex regex expressing where to cut see (see grep)
ignore.case whether or not regex should be case sensitive (see grep)
fixed whether or not regex should be interpreted as is or as regular expression (see grep)
perl whether or not Perl compatible regex should be used (see grep)
useBytes byte-by-byte matching of regex or character-by-character (see grep)
non_token should information for non-token, i.e. those patterns by which the text was split, be returned as well
Index

* datasets
  * prometheus_early, 3
  * prometheus_late, 3

* data
  * R6_rtext_extended, 4
  * rtext, 5
  * rtext_base, 7
  * rtext_export, 10
  * rtext_loadsave, 11
  * rtext_tokenize, 12

grep, 14

modus, 2

plot.rtext, 2
  * prometheus_early, 3
  * prometheus_late, 3

R6_rtext_extended, 4
  * R6Class, 4, 5, 7, 10–12
  * rtext, 5, 9, 11, 12, 14
  * rtext::R6_rtext_extended, 6, 7, 10–12
  * rtext::rtext_base, 6, 10–12
  * rtext::rtext_export, 6, 12
  * rtext::rtext_loadsave, 6, 10, 12
  * rtext::rtext_tokenize, 6
  * rtext_base, 7
  * rtext_export, 10
  * rtext_loadsave, 11
  * rtext_tokenize, 12

text_tokenize.rtext, 14