Package ‘rdracor’

May 9, 2026

Type Package

Title Access to the 'DraCor' API

Description Provide an interface for 'Drama Corpora Project' ('DraCor') API: <https://dracor.org/documentation/api>.

Version 1.0.6

License GPL (>= 3)

Encoding UTF-8

Imports jsonlite (>= 1.6), utils, data.table (>= 1.12.2), xml2 (>= 1.2.2), igraph (>= 1.2.4.1), httr (>= 1.4.1), tibble (>= 3.1.8), purrr (>= 0.3.5), stringr (>= 1.4.1), tidyr (>= 1.2.1), Rdpack (>= 2.4)

RoxygenNote 7.3.2

RdMacros Rdpack

Suggests testthat (>= 2.1.0)

URL https://github.com/dracor-org/rdracor

BugReports https://github.com/dracor-org/rdracor/issues

NeedsCompilation no

Author Ivan Pozdniakov [aut, cre] (ORCID: <https://orcid.org/0000-0002-2450-7004>)

Maintainer Ivan Pozdniakov <bucherr@yandex.ru>

Repository CRAN

Date/Publication 2025-09-29 16:20:02 UTC

Contents

`dracor_api` *Send a GET request to DraCor API and parse the results*

Description

Function `dracor_api()` sends a GET request to DraCor API with a specified expected type and parses results depending on selected expected type.

Usage

```
dracor_api(
  request,
 expected_type = c("application/json", "application/xml", "text/csv", "text/plain"),
  parse = TRUE,
  default_type = FALSE,
  split_text = TRUE,
  as_tibble = TRUE,
  ...
)
```

Arguments

`request`	Character, valid GET request.
`expected_type`	Character, 'MIME' type: one of `"application/json"`, `"application/xml"`, `"text/csv"`, `"text/plain"`.
`parse`	Logical, if `TRUE` (default value), then a response is parsed depending on `expected_type`. See details below.
`default_type`	Logical, if `TRUE`, default response data type is returned. Therefore, a response is not parsed and `parse` is ignored. The default value is `FALSE`.

split_text	Logical, if TRUE, plain text lines are read as different values in a vector instead of returning one character value. Default value is TRUE.
as_tibble	Logical, if TRUE, data frame will be returned as a tidyverse tibble (tbl_df). The default value is TRUE.
...	Other arguments passed to a parser function.

Details

There are four different 'MIME' types (aka internet media type) that can be retrieved for DraCor API, the specific combination of possible 'MIME' types depends on API command. When parse = TRUE is used, the content is parsed depending on selected 'MIME' type in expected_type:

application/json jsonlite::fromJSON()

application/xml xml2::read_xml()

text/csv data.table::fread()

text/plain No need for additional preprocessing

Value

A content of a response to GET method to the 'DraCor' API. If parse = FALSE or default_type = TRUE, a single character value is returned. Otherwise, the resulting value is parsed according to a value of default_type parameter. The resulting structure of the output depends on the selected default_type value, the respective function for parsing (see default_type) and additional parameters that are passed to the function for parsing.

See Also

dracor_sparql

Examples

```
dracor_api("https://dracor.org/api/v1/info", expected_type = "application/json")
```

---

dracor_api_info *Retrieve 'DraCor' API info*

Description

dracor_api_info() returns information about 'DraCor' API: name of the API, status, existdb version, API version etc.

Usage

```
dracor_api_info(dracor_api_url = NULL)

get_dracor_api_url()

set_dracor_api_url(new_dracor_api_url)
```

Arguments

dracor_api_url	Character, 'DraCor' API URL. If NULL (default), the current 'DraCor' API URL is used.
new_dracor_api_url	Character, 'DraCor' API URL that will replace the current 'DraCor' API URL.

Functions

- `get_dracor_api_url()`: Returns 'DraCor' API URL in use
- `set_dracor_api_url()`: Set new 'DraCor' API URL (globally), returns NULL

See Also

`dracor_api`

Examples

```
dracor_api_info()
dracor_api_info("https://staging.dracor.org/api")
get_dracor_api_url()
```

---

dracor_sparql *Submit SPARQL queries to DraCor API*

Description

`dracor_sparql()` submits SPARQL queries and parses the result.

Usage

```
dracor_sparql(sparql_query = NULL, parse = TRUE, ...)
```

Arguments

sparql_query	Character, SPARQL query.
parse	Logical, if TRUE the result is parsed by `xml2::read_xml()`, otherwise character value is returned. Default value is TRUE.
...	Additional arguments passed to `dracor_api`.

Value

SPARQL xml parsed.

See Also

`get_dracor`

Examples

```
dracor_sparql("SELECT * WHERE {?s ?p ?o} LIMIT 10")
# If you want to avoid parsing by xml2::read_xml():
dracor_sparql("SELECT * WHERE {?s ?p ?o} LIMIT 10", parse = FALSE)
```

---

get_character_plays *Retrieve plays having a character identified by 'Wikidata ID'*

Description

`get_character_plays()` requests plays that include a character that can by found in 'Wikidata' by it's id. `get_character_plays()` sends a request and parses the the result to get those plays as a data frame.

Usage

```
get_character_plays(char_wiki_id)
```

Arguments

`char_wiki_id`	Character value with 'Wikidata ID' for a character. 'Wikidata ID' can be found on https://www.wikidata.org/wiki/Wikidata:Main_Page. Character vector (longer than 1) is not supported.

Value

Data frame, in which one row represents one play. Information on author(s) name, character name, play name, URL and ID is represented in separate columns.

See Also

get_dracor

Examples

```
wiki_id <- "Q131412"
get_character_plays(wiki_id)
```

`get_dracor_meta` *Retrieve information on available corpora*

Description

`get_dracor_meta()` returns metadata on available corpora as a `dracor_meta` object that inherits data frame (and can be used as such). Use `summary()` and `plot()` on this object to get an even more condensed summary.

Usage

```
get_dracor_meta()

## S3 method for class 'dracor_meta'
summary(object, ...)

## S3 method for class 'dracor_meta'
plot(x, ...)
```

Arguments

`object`	An object of class "`dracor_meta`".
`...`	Other arguments to be passed.
`x`	A `dracor_meta` object.

Value

`dracor_meta` object that inherits data frame (and can be used as such).

Functions

- `summary(dracor_meta)`: Meaningful summary for `dracor_meta` object.
- `plot(dracor_meta)`: Plots how many plays are available for each corpus.

See Also

`get_dracor`

Examples

```
corpora_meta <- get_dracor_meta()
corpora_meta
summary(corpora_meta)
plot(corpora_meta)
```

`get_net_cooccur_edges` *Retrieve co-occurrence edges list for a play as a data frame*

Description

`get_net_cooccur_edges()` requests edges list for a play network, given corpus and play names. Each row represents co-occurrences of two characters in a play — number of scenes where two characters appeared together. This edges list can be used to construct a network for a play.

Usage

```
get_net_cooccur_edges(play = NULL, corpus = NULL, ...)

get_net_relations_edges(play = NULL, corpus = NULL, ...)
```

Arguments

`play`	Character, name of a play (you can find all play names in `"playName"` column within an object returned by `get_dracor`). Character vector (longer than 1) is not supported.
`corpus`	Character, name of the corpus (you can find all corpus names in `name` column within an object returned by `get_dracor_meta`).
`...`	Additional arguments passed to `dracor_api`.

Value

data frame with edges (each row = one edge of a network).

Functions

- `get_net_relations_edges()`: Retrieves kinship and other relationship data, following the encoding scheme proposed in (Wiedmer et al. 2020).

References

Wiedmer N, Pagel J, Reiter N (2020). "Romeo, Freund des Mercutio: Semi-Automatische Extraktion von Beziehungen zwischen dramatischen Figuren." In *Konferenz Digital Humanities im deutschsprachigen Raum*. doi:10.5281/zenodo.4621778.

See Also

`get_net_cooccur_igraph` `get_net_cooccur_gexf` `get_net_cooccur_graphml` `get_net_cooccur_metrics` `get_net_relations_igraph`

Examples

```
get_net_cooccur_edges(play = "lessing-emilia-galotti", corpus = "ger")
```

get_net_cooccur_gexf *Retrieve co-occurrence network for a play in 'GEXF'*

Description

get_net_cooccur_gexf() requests a play co-occurrence network in 'GEXF' (Graph Exchange XML Format), given play and corpus names. 'GEXF' is a format used in 'Gephi' — an open source software for network analysis and visualization.

Usage

```
get_net_cooccur_gexf(play = NULL, corpus = NULL, parse = TRUE, ...)

get_net_relations_gexf(play = NULL, corpus = NULL, ...)
```

Arguments

play	Character, name of a play (you can find all play names in "playName" column within an object returned by get_dracor). Character vector (longer than 1) is not supported.
corpus	Character, name of the corpus (you can find all corpus names in name column within an object returned by get_dracor_meta).
parse	Logical, if TRUE the result is parsed by xml2::read_xml(), otherwise character value is returned. Default value is TRUE.
...	Additional arguments passed to dracor_api.

Value

'GEXF' data.

Functions

- get_net_relations_gexf(): Retrieves kinship and other relationship data, following the encoding scheme proposed in (Wiedmer et al. 2020).

References

Wiedmer N, Pagel J, Reiter N (2020). "Romeo, Freund des Mercutio: Semi-Automatische Extraktion von Beziehungen zwischen dramatischen Figuren." In *Konferenz Digital Humanities im deutschsprachigen Raum*. doi:10.5281/zenodo.4621778.

See Also

get_net_cooccur_igraph get_net_cooccur_metrics get_net_cooccur_graphml get_net_cooccur_edges get_net_relations_igraph

Examples

```
get_net_cooccur_gexf(play = "lessing-emilia-galotti", corpus = "ger")
# If you want 'GEXF' without parsing by xml2::read_xml():
get_net_cooccur_gexf(
  play = "lessing-emilia-galotti",
  corpus = "ger",
  parse = FALSE
)
```

`get_net_cooccur_graphml` *Retrieve co-occurrence network for a play in 'GraphML'*

Description

`get_net_cooccur_graphml()` requests a play co-occurrence network in 'GraphML', given play and corpus names. 'GraphML' is a common format for graphs based on XML.

Usage

```
get_net_cooccur_graphml(play = NULL, corpus = NULL, parse = TRUE, ...)

get_net_relations_graphml(play = NULL, corpus = NULL, ...)
```

Arguments

`play`	Character, name of a play (you can find all play names in `"playName"` column within an object returned by `get_dracor`). Character vector (longer than 1) is not supported.
`corpus`	Character, name of the corpus (you can find all corpus names in `name` column within an object returned by `get_dracor_meta`).
`parse`	Logical, if `TRUE` the result is parsed by `xml2::read_xml()`, otherwise character value is returned. Default value is `TRUE`.
`...`	Additional arguments passed to `dracor_api`.

Value

'GraphML' data.

Functions

- `get_net_relations_graphml()`: Retrieves kinship and other relationship data, following the encoding scheme proposed in (Wiedmer et al. 2020).

References

Wiedmer N, Pagel J, Reiter N (2020). "Romeo, Freund des Mercutio: Semi-Automatische Extraktion von Beziehungen zwischen dramatischen Figuren." In *Konferenz Digital Humanities im deutschsprachigen Raum*. doi:10.5281/zenodo.4621778.

See Also

get_net_cooccur_igraph get_net_cooccur_gexf get_net_cooccur_metrics get_net_cooccur_edges get_net_relations_igraph

Examples

```
get_net_cooccur_graphml(play = "lessing-emilia-galotti", corpus = "ger")
# If you want 'GEXF' without parsing by xml2::read_xml():
get_net_cooccur_graphml(play = "lessing-emilia-galotti", corpus = "ger", parse = FALSE)
```

---

get_net_cooccur_igraph

Retrieve an igraph co-occurrence network for a play

Description

`get_net_cooccur_igraph()` returns a play network, given play and corpus names. Play network is constructed based on characters' co-occurrence matrix. Each node (vertex) is a character (circle) or a group of characters (square), edges width is proportional to the number of common play segments where two characters occur together.

Usage

```
get_net_cooccur_igraph(play = NULL, corpus = NULL, as_igraph = FALSE)

## S3 method for class 'cooccur_igraph'
plot(
  x,
  layout = igraph::layout_with_kk,
  vertex.label = label_cooccur_igraph(x),
  gender_colors = c(MALE = "#0073C2", FEMALE = "#EFC000", UNKNOWN = "#99979D"),
 vertex_size_metric = c("numOfWords", "numOfScenes", "numOfSpeechActs", "degree",
    "weightedDegree", "closeness", "betweenness", "eigenvector"),
  vertex_size_scale = c(5, 20),
  edge_size_scale = c(0.5, 4),
  vertex_label_adjust = TRUE,
  vertex.label.color = "#03070f",
  vertex.label.family = "sans",
  vertex.label.font = 2L,
  vertex.frame.color = "white",
```

```
  ...
)

## S3 method for class 'cooccur_igraph'
summary(object, ...)
```

Arguments

play	Character, name of a play (you can find all play names in "playName" column within an object returned by get_dracor). Character vector (longer than 1) is not supported.
corpus	Character, name of the corpus (you can find all corpus names in name column within an object returned by get_dracor_meta).
as_igraph	Logical, if TRUE, returns simple igraph object instead of cooccur_igraph. FALSE by default.
x	A cooccur_igraph object to plot.
layout	Function, an algorithm used for the graph layout. See igraph.plotting.
vertex.label	Character vector of character names. By default, function label_cooccur_igraph is used to avoid overplotting on large graphs.
gender_colors	Named vector with 3 values with colors for MALE, FEMALE and UNKNOWN respectively. Set NULL to use the default igraph colors. If you set parameter vertex.color (see igraph.plotting), gender_colors will be ignored.
vertex_size_metric	Character value, one of "numOfWords", "numOfScenes", "numOfSpeechActs", "degree", "weightedDegree", "closeness", "betweenness", "eigenvector" that will be used as a metric for vertex size. Alternatively, you can specify vertex size by yourself using parameter vertex.size(see igraph.plotting), in this case parameter vertex_size_metric is ignored.
vertex_size_scale	Numeric vector with two values. The first number is for mean size of node(vertex), the second one is for node size variance. If you specify vertex size by yourself using parameter vertex.size(see igraph.plotting), vertex_size_scale is ignored.
edge_size_scale	Numeric vector with two values. The first number defines average size of edges, the second number defines edges size variance. If you specify edges size by yourself using parameter edge.width(see igraph.plotting), edge_size_scale is ignored.
vertex_label_adjust	Logical. If TRUE, labels positions are moved to the top of the respectives nodes. If FALSE, labels are placed in the nodes centers. TRUE by default. If you set parameter vertex.label.dist(see igraph.plotting) by yourself, vertex_label_adjust is ignored.
vertex.label.color	See igraph.plotting.

vertex.label.family
: See igraph.plotting.

vertex.label.font
: See igraph.plotting.

vertex.frame.color
: See igraph.plotting.

...
: Other arguments to be passed to plot.igraph

object
: An object of class cooccur_igraph.

Value

cooccur_igraph — an object that inherits igraph and can be treated as such.

Functions

- plot(cooccur_igraph): Plot cooccur_igraph using plot.igraph with slightly modified defaults.
- summary(cooccur_igraph): Meaningful summary for "cooccur_igraph" object: network properties, gender distribution

See Also

get_net_relations_igraph label_cooccur_igraph

Examples

```
emilia_igraph <- get_net_cooccur_igraph(
  play = "lessing-emilia-galotti",
  corpus = "ger"
)
igraph::diameter(emilia_igraph)
plot(emilia_igraph)
summary(emilia_igraph)
```

---

get_net_cooccur_metrics

Retrieve co-occurrence network metrics for a play

Description

get_net_cooccur_metrics() requests network metrics for a specific play, given play and corpus names. Play network is constructed based on characters' co-occurrence matrix.

Usage

```
get_net_cooccur_metrics(play = NULL, corpus = NULL, ...)
```

Arguments

play	Character, name of a play (you can find all play names in "playName" column within an object returned by get_dracor). Character vector (longer than 1) is not supported.
corpus	Character, name of the corpus (you can find all corpus names in name column within an object returned by get_dracor_meta).
...	Additional arguments passed to dracor_api.

Value

List with network metrics for a specific play.

See Also

get_net_cooccur_igraph get_net_cooccur_gexf get_net_cooccur_graphml get_net_cooccur_edges get_net_relations_igraph

Examples

```
get_net_cooccur_metrics(play = "lessing-emilia-galotti", corpus = "ger")
```

---

get_net_relations_igraph

Retrieve an igraph relations network for a play

Description

get_net_relations_igraph() a play network, given play and corpus names . The network represent kinship and other relationships data, following the encoding scheme proposed in (Wiedmer et al. 2020).

Usage

```
get_net_relations_igraph(play = play, corpus = corpus, as_igraph = FALSE)

## S3 method for class 'relations_igraph'
summary(object, ...)

## S3 method for class 'relations_igraph'
plot(
  x,
  layout = igraph::layout_nicely,
  gender_colors = c(MALE = "#0073C2", FEMALE = "#EFC000", UNKNOWN = "#99979D"),
  show_others = c("vertex", "vertex_label", "none"),
  vertex_size = c(13, 4),
  vertex_label_size = c(0.8, 0.5),
```

```
  vertex_label_adjust = TRUE,
  vertex.label.color = "#03070f",
  vertex.label.family = "sans",
  vertex.label.font = 2L,
  vertex.frame.color = "white",
  edge.arrow.size = 0.25,
  edge.arrow.width = 1.5,
  edge.curved = 0.15,
  edge.label.family = "sans",
  edge.label.font = 4L,
  edge.label.cex = 0.75,
  ...
)
```

Arguments

play	Character, name of a play (you can find all play names in "playName" column within an object returned by get_dracor). Character vector (longer than 1) is not supported.
corpus	Character, name of the corpus (you can find all corpus names in name column within an object returned by get_dracor_meta).
as_igraph	Logical, if TRUE, returns simple igraph object instead of cooccur_igraph. FALSE by default.
object	An object of class relations_igraph.
...	Other arguments to be passed to plot.igraph
x	A relations_igraph object to plot.
layout	Function, an algorithm used for graph layout. See layout_.
gender_colors	Named vector with 3 values with colors for MALE, FEMALE and UNKNOWN respectively. Set NULL to use default igraph colors. If you set parameter vertex.color (see igraph.plotting), gender_colors will be ignored.
show_others	Character value. What to do with vertices without relations? • "vertex": plot only vertices without labels. • "vertex_label": plot both vertices and labels. • "none": do not plot vertices without relations. The default is "vertex".
vertex_size	Numeric vector with two values. The first number is for nodes with relations, the second number is for all other nodes.
vertex_label_size	Numeric vector with two values. The first number defines label sizes for nodes with relations, the second number for nodes without relations.
vertex_label_adjust	Logical value. If TRUE, labels positions are moved to the top of the respective nodes. If FALSE, labels are placed in the nodes centers. TRUE by default. If you set parameter vertex.label.dist(see igraph.plotting) by yourself, vertex_label_adjust is ignored.

vertex.label.color
: See igraph.plotting.

vertex.label.family
: See igraph.plotting.

vertex.label.font
: See igraph.plotting.

vertex.frame.color
: See igraph.plotting.

edge.arrow.size
: See igraph.plotting.

edge.arrow.width
: See igraph.plotting.

edge.curved
: See igraph.plotting.

edge.label.family
: See igraph.plotting.

edge.label.font
: See igraph.plotting.

edge.label.cex
: See igraph.plotting.

Value

`relations_igraph` — an object that inherits `igraph` and can be treated as such.

Functions

- `summary(relations_igraph)`: Meaningful summary for `"relations_igraph"` object: relationships and their type.
- `plot(relations_igraph)`: Plot `relations_igraph` using `plot.igraph` with slightly modified defaults.

References

Wiedmer N, Pagel J, Reiter N (2020). "Romeo, Freund des Mercutio: Semi-Automatische Extraktion von Beziehungen zwischen dramatischen Figuren." In *Konferenz Digital Humanities im deutschsprachigen Raum*. doi:10.5281/zenodo.4621778.

See Also

`get_net_cooccur_igraph`

Examples

```
galotti_relations <- get_net_relations_igraph(
  play = "lessing-emilia-galotti",
  corpus = "ger"
)
plot(galotti_relations)
summary(galotti_relations)
```

get_play_characters *Retrieve data for characters in a play*

Description

`get_play_characters()` requests miscellaneous information for characters in a play, given play and corpus names: name, number and size of their lines, gender, some network metrics etc.

Usage

```
get_play_characters(play = NULL, corpus = NULL, ...)
```

Arguments

play	Character, name of a play (you can find all play names in "playName" column within an object returned by get_dracor). Character vector (longer than 1) is not supported.
corpus	Character, name of the corpus (you can find all corpus names in name column within an object returned by get_dracor_meta).
...	Additional arguments passed to dracor_api.

Value

Data frame, every raw represents one character in the play.

See Also

get_play_metadata

Examples

```
get_play_characters(play = "lessing-emilia-galotti", corpus = "ger")
```

get_play_metadata *Retrieve metadata for a play*

Description

`get_play_metadata()` requests metadata for a specific play, given play and corpus names.

Usage

```
get_play_metadata(play = NULL, corpus = NULL, full_metadata = TRUE, ...)
```

Arguments

play	Character, name of a play (you can find all play names in "playName" column within an object returned by get_dracor). Character vector (longer than 1) is not supported.
corpus	Character, name of the corpus (you can find all corpus names in name column within an object returned by get_dracor_meta).
full_metadata	Logical: if TRUE (default value), then additional metadata are retrieved.
...	Additional arguments passed to dracor_api.

Value

List with the play metadata.

See Also

get_net_cooccur_edges get_play_rdf get_play_characters

Examples

```
get_play_metadata(
  play = "lessing-emilia-galotti",
  corpus = "ger",
  full_metadata = FALSE
)
```

---

get_play_rdf *Retrieve an RDF for a play*

Description

get_play_rdf() requests an RDF (Resource Description Framework) data for a play, given play and corpus names. RDF for plays can be useful for extraction data for a play from https://www.wikidata.org/wiki/Wikidata:Main_Page.

Usage

```
get_play_rdf(play = NULL, corpus = NULL, parse = TRUE, ...)
```

Arguments

play	Character, name of a play (you can find all play names in "playName" column within an object returned by get_dracor). Character vector (longer than 1) is not supported.
corpus	Character, name of the corpus (you can find all corpus names in name column within an object returned by get_dracor_meta).
parse	Logical, if TRUE the result is parsed by xml2::read_xml(), otherwise character value is returned. Default value is TRUE.
...	Additional arguments passed to dracor_api.

Value

RDF data parsed by xml2::read_xml().

See Also

get_play_metadata get_play_characters

Examples

```
get_play_rdf(play = "lessing-emilia-galotti", corpus = "ger")
# If you want RDF without parsing by xml2::read_xml():
get_play_rdf(play = "lessing-emilia-galotti", corpus = "ger", parse = FALSE)
```

---

get_text_chr_spoken *Retrieve lines and stage directions for a play*

Description

get_text_chr_spoken() request lines and stage directions for a play, given play and corpus names.

Usage

```
get_text_chr_spoken(
  play = NULL,
  corpus = NULL,
  gender = NULL,
  split_text = TRUE,
  ...
)

get_text_chr_spoken_bych(
  play = NULL,
  corpus = NULL,
  split_text = TRUE,
  as_data_frame = FALSE,
  ...
)

get_text_chr_stage(play = NULL, corpus = NULL, split_text = TRUE, ...)

get_text_chr_stage_with_sp(play = NULL, corpus = NULL, split_text = TRUE, ...)
```

Arguments

play	Character, name of a play (you can find all play names in "playName" column within an object returned by get_dracor). Character vector (longer than 1) is not supported.
corpus	Character, name of the corpus (you can find all corpus names in name column within an object returned by get_dracor_meta).
gender	Character, optional parameter to extract lines for characters of specified gender: "MALE", "FEMALE", "UNKNOWN".
split_text	If TRUE returns text as a character vector of lines. Otherwise, returns text as one character value. TRUE by default.
...	Additional arguments passed to dracor_api.
as_data_frame	If TRUE returns data frame with a row for every character and text in a column "text". Otherwise, a named list with character values is returned. FALSE by default.

Value

For get_text_chr_spoken(), get_text_chr_stage() and get_text_chr_stage_with_sp(): a character vector (if split_text = TRUE, the default value) or a single character value (if split_text = FALSE). For get_text_chr_spoken_bych():

split_text = TRUE **and** as_data_frame = FALSE **(default)** a named list with character vectors for every character

split_text = FALSE **and** as_data_frame = FALSE a named character vector (one value = one character)

split_text = TRUE **and** as_data_frame = TRUE a data frame: every row represent a character, text of a play is stored in a "text" column, the "text" column is a list column with a character vector of lines

split_text = FALSE **and** as_data_frame = TRUE a data frame: every row represent a character, text of a play is stored in a "text" column, the "text" column is a simple character column

Functions

- get_text_chr_spoken_bych(): Retrieves lines grouped by characters in a play, given play and corpus names.
- get_text_chr_stage(): Retrieves all stage directions of a play, given play and corpus names.
- get_text_chr_stage_with_sp(): Retrieves all stage directions of a play including speakers (if applicable), given play and corpus names.

See Also

get_text_tei get_text_df

Examples

```
get_text_chr_spoken(play = "lessing-emilia-galotti", corpus = "ger")
get_text_chr_spoken(
  play = "lessing-emilia-galotti",
  corpus = "ger",
  gender = "FEMALE"
)
get_text_chr_spoken(
  play = "lessing-emilia-galotti",
  corpus = "ger",
  gender = "FEMALE",
  split_text = FALSE
)
get_text_chr_spoken_bych(
  play = "lessing-emilia-galotti",
  corpus = "ger"
)
get_text_chr_stage(
  play = "lessing-emilia-galotti",
  corpus = "ger"
)
get_text_chr_stage_with_sp(
  play = "lessing-emilia-galotti",
  corpus = "ger"
)
```

---

get_text_tei *Retrieve a text for a play in 'TEI'*

Description

`get_text_tei()` requests a text for a play in 'TEI' format, given play and corpus names. 'TEI' is an XML vocabulary, which makes it easy to extract structural information (Fischer et al. 2019).

Usage

```
get_text_tei(play = NULL, corpus = NULL, ...)
```

Arguments

`play`	Character, name of a play (you can find all play names in `"playName"` column within an object returned by `get_dracor`). Character vector (longer than 1) is not supported.
`corpus`	Character, name of the corpus (you can find all corpus names in `name` column within an object returned by `get_dracor_meta`).
`...`	Additional arguments passed to `dracor_api`.

Value

TEI data parsed by `xml2::read_xml()`.

References

Fischer F, Börner I, Göbel M, Hechtl A, Kittel C, Milling C, Trilcke P (2019). "Programmable corpora: Introducing DraCor, an infrastructure for the research on European drama." In *Digital Humanities 2019: "Complexities" (DH2019)*. doi:10.5281/zenodo.4284002.

See Also

`get_text_df` `get_text_chr_spoken` `tei_to_df`

Examples

```
get_text_tei(play = "lessing-emilia-galotti", corpus = "ger")
# If you want a text in TEI without parsing by xml2::read_xml():
get_text_tei(play = "lessing-emilia-galotti", corpus = "ger", parse = FALSE)
```

---

label_cooccur_igraph *Extract labels for plotting 'cooccur_igraph' object*

Description

`label_cooccur_igraph()` returns labels for plotting `cooccur_igraph` object. `label_cooccur_igraph` gives control of overplotting for labels (i.e. character names) by deleting extra labels if there are too many of them. Thus, it highlights the most significant characters of the selected play. This function can be used to set `vertex.label` parameter for `plot.cooccur_igraph`.

Usage

```
label_cooccur_igraph(
  graph,
  max_graph_size = 30L,
  top_nodes = 3L,
 label_size_metric = c("betweenness", "numOfWords", "numOfScenes", "numOfSpeechActs",
    "degree", "weightedDegree", "closeness", "eigenvector")
)
```

Arguments

`graph`	`cooccur_igraph` object to plot.
`max_graph_size`	Integer, maximum network size for plotting all labels. If you don't want to delete any labels, set `Inf`.
`top_nodes`	Integer, number of labels to be plotted. Characters with the highest number of words will be selected.
`label_size_metric`	Character, a metric that is used to rank characters in a play.

Details

label_cooccur_igraph takes labels from a vertices data frame column "name", checks that network size is more than max_graph_size, if it is true, returns names for top top_nodes and NA for the rest.

Value

Character vector of character names.

See Also

get_net_cooccur_igraph

Examples

```
emilia_igraph <- get_net_cooccur_igraph(
  play = "lessing-emilia-galotti",
  corpus = "ger"
)
label_cooccur_igraph(emilia_igraph, max_graph_size = 10, top_nodes = 4)
```

---

summary.dracor *Retrieve metadata for all plays in selected corpora*

Description

get_dracor() request data on all plays in selected (or all) corpora. get_dracor() returns dracor object that inherits data frame (and can be used as such) but specified summary method.

Usage

```
## S3 method for class 'dracor'
summary(object, ...)

get_dracor(corpus = "all", full_metadata = TRUE)
```

Arguments

object	An object of class dracor.
...	Other arguments to be passed to summary.default.
corpus	Character vector with names of the corpora (you can find all corpora names in name column within an object returned by get_dracor_meta) or "all" (default value). if "all", then all available corpora are downloaded.
full_metadata	Logical: if TRUE (default value), then additional metadata are retrieved.

Details

You need to provide a vector with valid names of the corpora, e.g. "rus", "ger" or "shake". Use function get_dracor_meta to extract names for all available corpora.

Value

dracor object that inherits data frame (and can be used as such).

Functions

- summary(dracor): Meaningful summary for dracor_meta object.

See Also

get_dracor_meta

Examples

```
tat <- get_dracor("tat")
summary(tat)
get_dracor(c("ita", "span", "greek"))
get_dracor()
```

---

tei_to_df *Retrieve a text for a play as a data frame*

Description

The function get_text_df() returns you a data frame with text of the selected play. tei_to_df() allows to convert an existing 'TEI' object to a data frame.

Usage

```
tei_to_df(tei)

get_text_df(play, corpus)
```

Arguments

tei	A TEI object stored as an object of class xml_document. You can use this function if you have already downloaded TEI using get_text_tei.
play	Character, name of a play (you can find all play names in "playName" column within an object returned by get_dracor). Character vector (longer than 1) is not supported.
corpus	Character, name of the corpus (you can find all corpus names in name column within an object returned by get_dracor_meta).

Value

Text of a play as a data frame in tidy text format. Each row represent one token. The text tokenised by lines, notes and stage directions (<p>, <l>, <stage> or <note>). Column `text` contains text of the line, other columns contain metadata for the line.

Functions

- `get_text_df()`: Retrieves all stage directions of a play, given play and corpus names.

See Also

`get_play_metadata`

Examples

```
get_text_df(play = "lessing-emilia-galotti", corpus = "ger")
emilia_tei <- get_text_tei(play = "lessing-emilia-galotti", corpus = "ger")
tei_to_df(emilia_tei)
```

Index