

Package: pubmedDashboard (via r-universe)

August 28, 2024

Title Creating PubMed Data Visualization Dashboards

Version 0.0.3

Description Package to facilitate the creation of data visualization dashboards through the flexdashboard and easyPubMed packages. This package is now deprecated in favour of the pubDashboard package.

License GPL (>= 3)

Imports easyPubMed, rempsyc, insight, dplyr, rlang, purrr, stringr, stringi, countrycode, Ecfun, lubridate, tidy

Suggests ggplot2, RColorBrewer, DT, waffle (>= 1.0.1), ggflags (>= 0.0.3), plotly, ggtext, dygraphs, xts, rstudioapi, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.1

Depends R (>= 4.1.0)

LazyData true

URL <https://github.com/rempsyc/pubmedDashboard>,
<https://rempsyc.github.io/pubmedDashboard/>

BugReports <https://github.com/rempsyc/pubmedDashboard/issues>

Remotes jimjam-slam/ggflags, hrbrmstr/waffle

Repository <https://rempsyc.r-universe.dev>

RemoteUrl <https://github.com/rempsyc/pubmedDashboard>

RemoteRef HEAD

RemoteSha 823f2fb5ed2f606350181d914909155011340376

Contents

add_affiliation	3
add_region	3
all_articles_to_df	4
article_to_df2	5
batch_pubmed_download2	6
check_pubmed_api_token	7
clean_journals_continents	7
convert_hex_to_char	8
countries	8
detect_missing_journals	9
dygraph_year	9
extract_split_address	10
get_affiliation	10
get_country	11
journal_field	12
match_university	12
partial_vlookup	13
read_bind_all_data	14
render_dashboard	14
save_process_pubmed_batch	16
scatter_continent_year	17
scatter_country_year	18
scatter_figure1	19
table_continent	20
table_continent_journal	21
table_continent_year	22
table_country	22
table_country_journal	23
table_country_year	24
table_journal_count	25
table_missing_country	25
universities	26
us_states	26
waffle_continent	27
waffle_continent_journal	27
waffle_country	28
waffle_country_journal	29
world_capitals	30

add_affiliation	<i>Add affiliations to pubmedDashboard dataframe</i>
-----------------	--

Description

Add affiliations to pubmedDashboard dataframe

Usage

```
add_affiliation(data)
```

Arguments

data The dataframe on which to add affiliations (department and university).

Examples

```
## Not run:
pubmed_query_string <- paste(
  "passion [Title/Abstract]",
  "AND Dualistic Model of Passion [Text Word]",
  "AND ('2023/01/01' [Date - Publication] : '2023/12/31' [Date - Publication])"
)

d.fls <- batch_pubmed_download2(
  pubmed_query_string = pubmed_query_string,
  year_low = 2023,
  year_high = 2023
)
articles.df <- all_articles_to_df(d.fls)
articles.df2 <- add_affiliation(articles.df)
articles.df2[5, ]

## End(Not run)
```

add_region	<i>Add regions to pubmedDashboard dataframe</i>
------------	---

Description

Add regions to pubmedDashboard dataframe

Usage

```
add_region(data)
```

Arguments

data The dataframe on which to add region.

Examples

```
## Not run:
d.fls <- batch_pubmed_download2(
  pubmed_query_string = paste(
    "passion [Title/Abstract]",
    "AND Dualistic Model of Passion [Text Word]",
    "AND ('2023/01/01' [Date - Publication] : '2023/12/31' [Date - Publication])"
  ),
  year_low = 2023,
  year_high = 2023
)
articles.df <- all_articles_to_df(d.fls)
articles.df2 <- add_affiliation(articles.df)
articles.df3 <- match_university(articles.df2)
articles.df4 <- add_region(articles.df3)
articles.df4[2, ]

## End(Not run)
```

all_articles_to_df *Convert list of PubMed XLM files to dataframe*

Description

Convert list of PubMed XLM files to dataframe

Usage

```
all_articles_to_df(d.fls)
```

Arguments

d.fls The list of XLM PubMed data.

Examples

```
## Not run:
pubmed_query_string <- paste(
  "passion [Title/Abstract]",
  "AND Dualistic Model of Passion [Text Word]",
  "AND ('2023/01/01' [Date - Publication] : '2023/12/31' [Date - Publication])"
)
d.fls <- batch_pubmed_download2(
  pubmed_query_string = pubmed_query_string,
```

```

    year_low = 2023,
    year_high = 2023
  )
  articles.df <- all_articles_to_df(d.files)
  articles.df[5, ]

## End(Not run)

```

article_to_df2 *Convert PubMed XLM files to dataframe*

Description

Convert PubMed XLM files to dataframe

Usage

```

article_to_df2(
  pubmedArticle,
  autofill = FALSE,
  max_chars = 500,
  getKeywords = FALSE,
  getAuthors = TRUE
)

```

Arguments

pubmedArticle	The PubMed data, in XLM form.
autofill	Autofill the affiliation address when missing.
max_chars	Maximum number of characters for the address.
getKeywords	Whether to extract keywords as well.
getAuthors	Whether to extract the author information.

Examples

```

## Not run:
dami_query <- paste(
  "passion [Title/Abstract]",
  "AND Dualistic Model of Passion [Text Word]",
  "AND ('2023/01/01' [Date - Publication] : '2023/12/31' [Date - Publication])"
)
dami_on_pubmed <- easyPubMed::get_pubmed_ids(dami_query)
dami_abstracts_xml <- easyPubMed::fetch_pubmed_data(dami_on_pubmed)
dami_abstracts_list <- easyPubMed::articles_to_list(dami_abstracts_xml)
article_to_df2(pubmedArticle = dami_abstracts_list[[2]], autofill = FALSE)[1, ]

## End(Not run)

```

`batch_pubmed_download2`*Download PubMed data with query string*

Description

Download PubMed data with query string

Usage

```
batch_pubmed_download2(  
  pubmed_query_string,  
  year_low = 2023,  
  year_high = 2030,  
  data_folder = "data",  
  batch_size = 5000,  
  api_key = NULL  
)
```

Arguments

<code>pubmed_query_string</code>	The PubMed query string.
<code>year_low</code>	The year the data should start.
<code>year_high</code>	The year the data should end.
<code>data_folder</code>	Where to save the data.
<code>batch_size</code>	The download batch size.
<code>api_key</code>	The api key for faster processing (optional).

Examples

```
## Not run:  
pubmed_query_string <- paste(  
  "passion [Title/Abstract]",  
  "AND Dualistic Model of Passion [Text Word]"  
)  
  
batch_pubmed_download2(  
  pubmed_query_string,  
  year_low = 2023,  
  year_high = 2023  
)  
  
## End(Not run)
```

`convert_hex_to_char` *Convert a Hex string to regular text*

Description

Convert a Hex string to regular text

Usage

```
convert_hex_to_char(hex_string)
```

Arguments

`hex_string` The address containg hex characters to convert.

Examples

```
address <- c(
  "D&#xe9;partement de Psychologie, Universit&#xe9; du Qu&#xe9;bec &#xe0; Montr&#xe9;al",
  "Department of Behavioural and Cognitive Sciences, University of Luxembourg"
)
convert_hex_to_char(address)
```

`countries` *List of countries taken from the package countrycode*

Description

List of countries taken from the package countrycode

Usage

```
countries
```

Format

A vector of class character containing country names.

detect_missing_journals
Detect missing journals

Description

Detect missing journals

Usage

```
detect_missing_journals(data)
```

Arguments

data The processed dataframe of data

dygraph_year *Generate a dygraph of journal paper percentages, by country and year*

Description

Generate a dygraph of journal paper percentages, by country and year

Usage

```
dygraph_year(data, level = "continent")
```

Arguments

data The processed dataframe of data
level Level of analysis, either country or continent

Examples

```
## Not run:  
pubmed_query_string <- paste(  
  "passion [Title/Abstract]",  
  "AND Dualistic Model of Passion [Text Word]"  
)  
  
save_process_pubmed_batch(  
  pubmed_query_string,  
  year_low = 2021,  
  year_high = 2023  
)  
data <- read_bind_all_data()
```

```
dygraph_year(data)
dygraph_year(data, "country")

## End(Not run)
```

`extract_split_address` *Extracts correct component from a splitted affiliation address*

Description

Extracts correct component from a splitted affiliation address

Usage

```
extract_split_address(splitted.address, string)
```

Arguments

`splitted.address`
The address splitted with `stringr::str_split`.

`string`
The keyword to determine which string component to keep.

Examples

```
address <- c(
  "Department of Psychology",
  " Cornell University",
  " Ithaca",
  " New York 14853-7601."
)

extract_split_address(address, "University")
extract_split_address(address, "Department")
```

`get_affiliation` *Get affiliations*

Description

Get affiliations

Usage

```
get_affiliation(address, info = "university")
```

Arguments

address The address to parse.
 info The information to extract, one of c("university", "department").

Examples

```
address <- c(
  "Department of Psychology, Cornell University, Ithaca, New York 14853-7601.",
  "Dipartimento di Psicologia Generale, Università di Padova, Italy.",
  "Universität Mannheim, Federal Republic of Germany.",
  "Département de psychologie, Université du Québec à Montréal, Canada."
)

get_affiliation(address, "department")
get_affiliation(address, "university")
```

get_country	<i>Get country</i>
-------------	--------------------

Description

Get country

Usage

```
get_country(address)
```

Arguments

address The address to parse.

Examples

```
## Not run:
d.fls <- batch_pubmed_download2(
  pubmed_query_string = paste(
    "passion [Title/Abstract]",
    "AND Dualistic Model of Passion [Text Word]",
    "AND ('2023/01/01' [Date - Publication] : '2023/12/31' [Date - Publication])"
  ),
  year_low = 2023,
  year_high = 2023
)
articles.df <- all_articles_to_df(d.fls)
articles.df2 <- add_affiliation(articles.df)
articles.df3 <- match_university(articles.df2)
get_country(articles.df3$address)
```

```
## End(Not run)
```

journal_field	<i>List of academic journals and corresponding fields</i>
---------------	---

Description

List of academic journals and corresponding fields

Usage

```
journal_field
```

Format

A data frame with 25 rows and 3 variables:

journal academic journal

journal_short short name for the journal

field the field of research

original_journal Whether it is part of one of the six original journals ...

match_university	<i>Match list of universities to countries in a dataframe</i>
------------------	---

Description

Match list of universities to countries in a dataframe

Usage

```
match_university(data)
```

Arguments

data The dataframe to use for matching.

Examples

```
## Not run:
d.fls <- batch_pubmed_download2(
  pubmed_query_string = paste(
    "passion [Title/Abstract]",
    "AND Dualistic Model of Passion [Text Word]",
    "AND ('2023/01/01' [Date - Publication] : '2023/12/31' [Date - Publication])"
  )
)
articles.df <- all_articles_to_df(d.fls)
articles.df2 <- add_affiliation(articles.df)
articles.df3 <- match_university(articles.df2)
articles.df3[5, ]

## End(Not run)
```

partial_vlookup *Like Excel vlookup/grep, both both ways*

Description

Like Excel vlookup/grep, both both ways

Usage

```
partial_vlookup(pattern, lookup_vector)
```

Arguments

pattern The pattern to compare.
lookup_vector The dictionary in which to look for the pattern.

Examples

```
address <- c(
  "Department of Psychology, Cornell University, Ithaca, New York 14853-7601.",
  "Dipartimento di Psicologia Generale, Università di Padova, Italy.",
  "Universität Mannheim, Federal Republic of Germany.",
  "Département de psychologie, Université du Québec à Montréal, Canada."
)
partial_vlookup(address, universities$university)
uni <- c(
  "Cornell University", "Università di Padova",
  "Universität Mannheim", "Université du Québec à Montréal"
)
partial_vlookup(uni, universities$university)
```

read_bind_all_data	<i>Read local pubmedDashboard data files and bind them in a single dataframe</i>
--------------------	--

Description

Read local pubmedDashboard data files and bind them in a single dataframe

Usage

```
read_bind_all_data(data_folder = "data")
```

Arguments

data_folder	The folder in which the data lives
-------------	------------------------------------

render_dashboard	<i>Render complete pubmedDashboard dashboard</i>
------------------	--

Description

Render complete pubmedDashboard dashboard

Usage

```
render_dashboard(  
  file_name = "dashboard",  
  title = "title",  
  author = "author",  
  pubmed_query_string = "",  
  journal = NULL,  
  year_low = 2023,  
  year_high = 2023,  
  month_low = "01",  
  month_high = 12,  
  day_low = "01",  
  day_high = 31,  
  data_folder = "data",  
  batch_size = 5000,  
  api_key = NULL,  
  verbose = TRUE,  
  query_pubmed = FALSE,  
  tab_continent = TRUE,  
  tab_continent_year = TRUE,  
  tab_continent_journal = TRUE,
```

```

    tab_country = TRUE,
    tab_country_journal = TRUE,
    tab_psychology = FALSE,
    tab_economics = FALSE,
    tab_general = FALSE,
    tab_figure1 = FALSE,
    tab_missing = TRUE
)

```

Arguments

file_name	Desired file name.
title	Desired dashboard title.
author	Desired displayed dashboard author.
pubmed_query_string	The PubMed query string.
journal	The list of desired journals.
year_low	The year the data should start.
year_high	The year the data should end.
month_low	The year the data should start.
month_high	The year the data should end.
day_low	The year the data should start.
day_high	The year the data should end.
data_folder	Where to save the data.
batch_size	The download batch size.
api_key	The api key for faster processing (optional).
verbose	Whether to include progress messages.
query_pubmed	Whether to query pubmed. This must be set to TRUE explicitly to avoid long operations. When the data is already downloaded and available in the data folder, this step is unnecessary.
tab_continent	Whether to render the "Continent" tab.
tab_continent_year	Whether to render the "Continent by year" tab.
tab_continent_journal	Whether to render the "Continent by journal" tab.
tab_country	Whether to render the "Country" tab.
tab_country_journal	Whether to render the "Country by journal" tab.
tab_psychology	Whether to render the "Psychology" tab.
tab_economics	Whether to render the "Economics" tab.
tab_general	Whether to render the "General" tab.
tab_figure1	Whether to render the "Figure 1" tab.
tab_missing	Whether to render the "Missing" tab.

Examples

```
## Not run:
render_dashboard(
  file_name = "my_dashboard",
  title = "Wonderful Dashboard",
  author = "Rémi Thériault",
  pubmed_query_string = "passion [Title/Abstract]",
  journal = c("Journal of Personality and Social Psychology", "Health Psychology"),
  year_low = 2023,
  year_high = 2023,
  query_pubmed = TRUE,
  tab_figure1 = TRUE
)

## End(Not run)
```

```
save_process_pubmed_batch
```

Mega function to process and save PubMed data

Description

Mega function to process and save PubMed data

Usage

```
save_process_pubmed_batch(
  pubmed_query_string = "",
  journal = NULL,
  year_low = 2024,
  year_high = 2024,
  month_low = "01",
  month_high = 12,
  day_low = "01",
  day_high = 31,
  data_folder = "data",
  suffix = "",
  batch_size = 5000,
  api_key = NULL,
  verbose = TRUE
)
```

Arguments

pubmed_query_string The PubMed query string.

journal The list of desired journals.

year_low	The year the data should start.
year_high	The year the data should end.
month_low	The year the data should start.
month_high	The year the data should end.
day_low	The year the data should start.
day_high	The year the data should end.
data_folder	Where to save the data.
suffix	What suffix to add to the name file.
batch_size	The download batch size.
api_key	The api key for faster processing (optional).
verbose	Whether to include progress messages.

Examples

```
## Not run:
pubmed_query_string <- paste(
  "passion [Title/Abstract]",
  "AND Dualistic Model of Passion [Text Word]"
)

save_process_pubmed_batch(
  pubmed_query_string,
  year_low = 2023,
  year_high = 2023
)

## End(Not run)
```

scatter_continent_year

Generate table of journal paper percentages, by continent and year

Description

Generate table of journal paper percentages, by continent and year

Usage

```
scatter_continent_year(
  data,
  method = "lm",
  plotly = TRUE,
  citation = NULL,
  citation_size = 15,
  ...
)
```

Arguments

data	The processed dataframe of data
method	Which method to use for the regression line, either "lm" (default) or "loess".
plotly	Logical, whether to use plotly for dynamic data visualization.
citation	Optionally, a citation to add as a footer.
citation_size	Font size of the citation.
...	Further arguments passed to rempsyc::nice_scatter

Examples

```
## Not run:
pubmed_query_string <- paste(
  "passion [Title/Abstract]",
  "AND Dualistic Model of Passion [Text Word]"
)

save_process_pubmed_batch(
  pubmed_query_string,
  year_low = 2022,
  year_high = 2023
)

data <- read_bind_all_data()
suppressWarnings(scatter_continent_year(data))

## End(Not run)
```

scatter_country_year *Generate table of journal paper percentages, by continent and year*

Description

Generate table of journal paper percentages, by continent and year

Usage

```
scatter_country_year(
  data,
  method = "lm",
  plotly = TRUE,
  citation = NULL,
  citation_size = 15,
  ...
)
```

Arguments

data	The processed dataframe of data
method	Which method to use for the regression line, either "lm" (default) or "loess".
plotly	Logical, whether to use plotly for dynamic data visualization.
citation	Optionally, a citation to add as a footer.
citation_size	Font size of the citation.
...	Further arguments passed to rempsyc::nice_scatter

Examples

```
## Not run:
pubmed_query_string <- paste(
  "passion [Title/Abstract]",
  "AND Dualistic Model of Passion [Text Word]"
)

save_process_pubmed_batch(
  pubmed_query_string,
  year_low = 2018,
  year_high = 2020
)
data <- read_bind_all_data()
suppressWarnings(scatter_country_year(data))

## End(Not run)
```

scatter_figure1	<i>Generate table of journal paper percentages, by continent and year</i>
-----------------	---

Description

Generate table of journal paper percentages, by continent and year

Usage

```
scatter_figure1(data, method = "lm", original = TRUE, plotly = TRUE, ...)
```

Arguments

data	The processed dataframe of data
method	Which method to use for the regression line, either "lm" (default) or "loess".
original	Logical; if TRUE, attempts to mimic Arnett's (2008) Figure 1 in style.
plotly	Logical, whether to use plotly for dynamic data visualization.
...	Further arguments passed to rempsyc::nice_scatter

Examples

```
## Not run:
pubmed_query_string <- paste(
  "Developmental Psychology [Journal]",
  "OR Journal of Personality and Social Psychology [Journal]",
  "OR Journal of Abnormal Psychology [Journal]",
  "OR Journal of Family Psychology [Journal]",
  "OR Health Psychology [Journal]",
  "OR Journal of Educational Psychology [Journal]"
)

save_process_pubmed_batch(
  pubmed_query_string,
  year_low = 2023,
  year_high = 2023
)
data <- read_bind_all_data()
scatter_figure1(data)

## End(Not run)
```

table_continent	<i>Generate table of journal paper percentages, by continent</i>
-----------------	--

Description

Generate table of journal paper percentages, by continent

Usage

```
table_continent(data, datatable = TRUE)
```

Arguments

data	The processed dataframe of data
datatable	Whether to output a DT::datatable HTML table widget instead of a regular dataframe (defaults to TRUE).

Examples

```
## Not run:
pubmed_query_string <- paste(
  "passion [Title/Abstract]",
  "AND Dualistic Model of Passion [Text Word]"
)

save_process_pubmed_batch(
  pubmed_query_string,
```

```
    year_low = 2023,  
    year_high = 2023  
  )  
  data <- read_bind_all_data()  
  table_continent(data)  
  
## End(Not run)
```

table_continent_journal

Generate table of journal paper percentages, by continent and journals

Description

Generate table of journal paper percentages, by continent and journals

Usage

```
table_continent_journal(data, datatable = TRUE)
```

Arguments

data	The processed dataframe of data
datatable	Whether to output a DT::datatable HTML table widget instead of a regular dataframe (defaults to TRUE).

Examples

```
## Not run:  
pubmed_query_string <- paste(  
  "passion [Title/Abstract]",  
  "AND Dualistic Model of Passion [Text Word]"  
)  
  
save_process_pubmed_batch(  
  pubmed_query_string,  
  year_low = 2023,  
  year_high = 2023  
)  
data <- read_bind_all_data()  
table_continent_journal(data)  
  
## End(Not run)
```

table_continent_year *Generate table of journal paper percentages, by continent and year*

Description

Generate table of journal paper percentages, by continent and year

Usage

```
table_continent_year(data, datatable = TRUE)
```

Arguments

data	The processed dataframe of data
datatable	Whether to output a DT::datatable HTML table widget instead of a regular dataframe (defaults to TRUE).

Examples

```
## Not run:
pubmed_query_string <- paste(
  "passion [Title/Abstract]",
  "AND Dualistic Model of Passion [Text Word]"
)

save_process_pubmed_batch(
  pubmed_query_string,
  year_low = 2022,
  year_high = 2023
)
data <- read_bind_all_data()
table_continent_year(data)

## End(Not run)
```

table_country *Generate table of journal paper percentages, by country*

Description

Generate table of journal paper percentages, by country

Usage

```
table_country(data, datatable = TRUE)
```

Arguments

data	The processed dataframe of data
datatable	Whether to output a DT::datatable HTML table widget instead of a regular dataframe (defaults to TRUE).

Examples

```
## Not run:
pubmed_query_string <- paste(
  "passion [Title/Abstract]",
  "AND Dualistic Model of Passion [Text Word]"
)

save_process_pubmed_batch(
  pubmed_query_string,
  year_low = 2023,
  year_high = 2023
)
data <- read_bind_all_data()
table_country(data)

## End(Not run)
```

table_country_journal *Generate table of journal paper percentages, by continent and year*

Description

Generate table of journal paper percentages, by continent and year

Usage

```
table_country_journal(data, datatable = TRUE)
```

Arguments

data	The processed dataframe of data
datatable	Whether to output a DT::datatable HTML table widget instead of a regular dataframe (defaults to TRUE).

Examples

```
## Not run:
pubmed_query_string <- paste(
  "passion [Title/Abstract]",
  "AND Dualistic Model of Passion [Text Word]"
)
)
```

```
save_process_pubmed_batch(  
  pubmed_query_string,  
  year_low = 2023,  
  year_high = 2023  
)  
data <- read_bind_all_data()  
table_country_journal(data)  
  
## End(Not run)
```

table_country_year *Generate table of journal paper percentages, by country and year*

Description

Generate table of journal paper percentages, by country and year

Usage

```
table_country_year(data, datatable = TRUE)
```

Arguments

data	The processed dataframe of data
datatable	Whether to output a DT::datatable HTML table widget instead of a regular dataframe (defaults to TRUE).

Examples

```
## Not run:  
pubmed_query_string <- paste(  
  "passion [Title/Abstract]",  
  "AND Dualistic Model of Passion [Text Word]"  
)  
  
save_process_pubmed_batch(  
  pubmed_query_string,  
  year_low = 2023,  
  year_high = 2023  
)  
data <- read_bind_all_data()  
table_country_year(data)  
  
## End(Not run)
```

table_journal_count	<i>Count number of papers per journal, with year range</i>
---------------------	--

Description

Count number of papers per journal, with year range

Usage

```
table_journal_count(data, datatable = TRUE)
```

Arguments

data	The processed dataframe of data
datatable	Whether to output a DT::datatable HTML table widget instead of a regular dataframe (defaults to TRUE).

table_missing_country	<i>Generate table of journal paper percentages, by country</i>
-----------------------	--

Description

Generate table of journal paper percentages, by country

Usage

```
table_missing_country(data, datatable = TRUE)
```

Arguments

data	The processed dataframe of data
datatable	Whether to output a DT::datatable HTML table widget instead of a regular dataframe (defaults to TRUE).

Examples

```
## Not run:
pubmed_query_string <- paste(
  "passion [Title/Abstract]",
  "AND Dualistic Model of Passion [Text Word]"
)

save_process_pubmed_batch(
  pubmed_query_string,
  year_low = 2023,
  year_high = 2023
```

```

)
data <- read_bind_all_data()
data[1, c(4, 6)] <- NA
table_missing_country(data)

## End(Not run)

```

universities	<i>A data frame of university and corresponding country</i>
--------------	---

Description

Obtained from GitHub, and then modified with minor improvements and more universities.

Usage

```
universities
```

Format

A data frame with 9420 rows and 2 variables:

country_code the country code

university the university ...

Source

<https://raw.githubusercontent.com/endSly/world-universities-csv/master/world-universities.csv>

us_states	<i>List of US states taken from the package countrycode</i>
-----------	---

Description

List of US states taken from the package countrycode

Usage

```
us_states
```

Format

A data frame with 50 rows and 3 variables:

state.name the name of the state

state.abb the name of the abbreviation

state.regex the regex for that state ...

waffle_continent	<i>Generate a waffle chart of journal paper percentages, by continent (each square = 1% of data)</i>
------------------	--

Description

Generate a waffle chart of journal paper percentages, by continent (each square = 1% of data)

Usage

```
waffle_continent(data, citation = NULL, citation_size = NULL)
```

Arguments

data	The processed dataframe of data
citation	Optionally, a citation to add as a footer.
citation_size	Font size of the citation.

Examples

```
## Not run:
pubmed_query_string <- paste(
  "passion [Title/Abstract]",
  "AND Dualistic Model of Passion [Text Word]"
)

save_process_pubmed_batch(
  pubmed_query_string,
  year_low = 2023,
  year_high = 2023
)
data <- read_bind_all_data()
waffle_continent(data)

## End(Not run)
```

waffle_continent_journal	<i>Generate a waffle chart of journal paper percentages, by continent (each square = 1% of data)</i>
--------------------------	--

Description

Generate a waffle chart of journal paper percentages, by continent (each square = 1% of data)

Usage

```
waffle_continent_journal(  
  data,  
  citation,  
  citation_size = NULL,  
  journal_abbreviation = TRUE  
)
```

Arguments

data	The processed dataframe of data
citation	Optionally, a citation to add as a footer.
citation_size	Font size of the citation.
journal_abbreviation	Logical, whether to use the journal abbreviation to fit the entire plot, otherwise some journal names can be quite long and accordingly be cropped.

Examples

```
## Not run:  
pubmed_query_string <- paste(  
  "passion [Title/Abstract]",  
  "AND Dualistic Model of Passion [Text Word]"  
)  
  
save_process_pubmed_batch(  
  pubmed_query_string,  
  year_low = 2023,  
  year_high = 2023  
)  
data <- read_bind_all_data()  
waffle_continent_journal(data)  
  
## End(Not run)
```

waffle_country

Generate a waffle plot made of country flags

Description

Generate a waffle plot made of country flags

Usage

```
waffle_country(data, citation, citation_size = NULL)
```

Arguments

data The processed dataframe of data
 citation Optionally, a citation to add as a footer.
 citation_size Font size of the citation.

Examples

```

## Not run:
pubmed_query_string <- paste(
  "passion [Title/Abstract]",
  "AND Dualistic Model of Passion [Text Word]"
)

save_process_pubmed_batch(
  pubmed_query_string,
  year_low = 2023,
  year_high = 2023
)
data <- read_bind_all_data()
waffle_country(data)

## End(Not run)

```

waffle_country_journal

*Generate a waffle chart of journal paper percentages, by continent
(each square = 1% of data)*

Description

Generate a waffle chart of journal paper percentages, by continent (each square = 1% of data)

Usage

```

waffle_country_journal(
  data,
  citation,
  citation_size = NULL,
  journal_abbreviation = TRUE
)

```

Arguments

data The processed dataframe of data
 citation Optionally, a citation to add as a footer.
 citation_size Font size of the citation.

```
journal_abbreviation
```

Logical, whether to use the journal abbreviation to fit the entire plot, otherwise some journal names can be quite long and accordingly be cropped.

Examples

```
## Not run:
pubmed_query_string <- paste(
  "passion [Title/Abstract]",
  "AND Dualistic Model of Passion [Text Word]"
)

save_process_pubmed_batch(
  pubmed_query_string,
  year_low = 2023,
  year_high = 2023
)
data <- read_bind_all_data()
waffle_country_journal(data)

## End(Not run)
```

world_capitals

List of world capitals taken from the package maps

Description

List of world capitals taken from the package maps

Usage

```
world_capitals
```

Format

A data frame with 259 rows and 6 variables:

name the name of the capital

country.etc the country of the capital

pop population of the capital

lat latitude of the capital

long longitude of the capital

capital whether it is a capital ...

Index

* datasets

- countries, 8
- journal_field, 12
- universities, 26
- us_states, 26
- world_capitals, 30

- add_affiliation, 3
- add_region, 3
- all_articles_to_df, 4
- article_to_df2, 5

- batch_pubmed_download2, 6

- check_pubmed_api_token, 7
- clean_journals_continents, 7
- convert_hex_to_char, 8
- countries, 8

- detect_missing_journals, 9
- DT::datatable, 20–25
- dygraph_year, 9

- extract_split_address, 10

- get_affiliation, 10
- get_country, 11

- journal_field, 12

- match_university, 12

- partial_vlookup, 13

- read_bind_all_data, 14
- rempsyc::nice_scatter, 18, 19
- render_dashboard, 14

- save_process_pubmed_batch, 16
- scatter_continent_year, 17
- scatter_country_year, 18
- scatter_figure1, 19

- stringr::str_split, 10

- table_continent, 20
- table_continent_journal, 21
- table_continent_year, 22
- table_country, 22
- table_country_journal, 23
- table_country_year, 24
- table_journal_count, 25
- table_missing_country, 25

- universities, 26
- us_states, 26

- waffle_continent, 27
- waffle_continent_journal, 27
- waffle_country, 28
- waffle_country_journal, 29
- world_capitals, 30