Package ‘rbioapi’

March 30, 2024

Type Package

Title User-Friendly R Interface to Biologic Web Services' API

Version 0.8.1

Description Currently fully supports Enrichr, JASPAR, miEAA, PANTHER, Reactome, STRING, and UniProt! The goal of rbioapi is to provide a user-friendly and consistent interface to biological databases and services. In a way that insulates the user from the technicalities of using web services API and creates a unified and easy-to-use interface to biological and medical web services. This is an ongoing project; New databases and services will be added periodically. Feel free to suggest any databases or services you often use.

License GPL-3


BugReports https://github.com/moosa-r/rbioapi/issues

Imports httr, jsonlite, utils

Suggests DT, knitr, png, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

RoxygenNote 7.3.1

NeedsCompilation no

Author Moosa Rezwani [aut, cre] (<https://orcid.org/0000-0001-6325-4444>)

Maintainer Moosa Rezwani <moosa.rezwani@gmail.com>

Repository CRAN

Date/Publication 2024-03-30 04:50:03 UTC
R topics documented:

rba_connection_test ............................................. 4
rba_enrichr ....................................................... 5
rba_enrichr_add_list ........................................... 7
rba_enrichr_enrich .............................................. 9
rba_enrichr_gene_map .......................................... 11
rba_enrichr_libs .................................................. 12
rba_enrichr_view_list ......................................... 13
rba_jaspar_collections ........................................ 15
rba_jaspar_collections_matrices ............................... 16
rba_jaspar_matrix ................................................ 18
rba_jaspar_matrix_search ....................................... 19
rba_jaspar_matrix_versions .................................. 21
rba_jaspar_releases ............................................. 23
rba_jaspar_sites .................................................. 24
rba_jaspar_species .............................................. 25
rba_jaspar_species_matrices .................................. 26
rba_jaspar_taxons ............................................... 28
rba_jaspar_taxons_matrices .................................... 29
rba_jaspar_tffm ................................................... 31
rba_jaspar_tffm_search .......................................... 32
rba_mieaa_cats .................................................... 34
rba_mieaa_convert_type ......................................... 35
rba_mieaa_convert_version ..................................... 37
rba_mieaa_enrich ................................................ 38
rba_mieaa_enrich_results ...................................... 41
rba_mieaa_enrich_status ....................................... 42
rba_mieaa_enrich_submit ....................................... 43
rba_options ........................................................ 46
rba_pages .......................................................... 48
rba_panther_enrich .............................................. 49
rba_panther_family ............................................... 51
rba_panther_homolog ............................................ 52
rba_panther_info .................................................. 53
rba_panther_mapping ............................................. 55
rba_panther_ortholog ............................................ 56
rba_panther_tree_grafter ....................................... 57
rba_reactome_analysis .......................................... 59
rba_reactome_analysis_download ............................... 62
rba_reactome_analysis_import ................................ 64
rba_reactome_analysis_mapping ................................. 65
rba_reactome_analysis_pdf ..................................... 67
rba_reactome_analysis_species ................................ 69
rba_reactome_analysis_token .................................. 71
rba_reactome_complex_list ..................................... 73
rba_reactome_complex_subunits ................................. 74
rba_reactome_diseases .......................................... 76
R topics documented:

- rba_reactome_entity_other_forms ................................................. 77
- rba_reactome_event_ancestors .................................................. 78
- rba_reactome_event_hierarchy .................................................... 80
- rba_reactome_exporter_diagram .................................................. 81
- rba_reactome_exporter_event ..................................................... 84
- rba_reactome_exporter_overview ............................................... 86
- rba_reactome_exporter_reaction ............................................... 88
- rba_reactome_interactors_psicquic ........................................... 90
- rba_reactome_interactors_static .............................................. 92
- rba_reactome_mapping ............................................................ 94
- rba_reactome_orthology .......................................................... 95
- rba_reactome_participants ....................................................... 97
- rba_reactome_participant_of .................................................... 98
- rba_reactome_pathways_events ............................................... 100
- rba_reactome_pathways_low ..................................................... 101
- rba_reactome_pathways_top ..................................................... 103
- rba_reactome_people_id .......................................................... 104
- rba_reactome_people_name ..................................................... 106
- rba_reactome_query ............................................................... 107
- rba_reactome_species ............................................................. 108
- rba_reactome_version ............................................................. 110
- rba_reactome_xref ................................................................. 111
- rba_string_annotations .......................................................... 112
- rba_string_enrichment ............................................................ 114
- rba_string_enrichment_ppi ....................................................... 115
- rba_string_homology_inter ...................................................... 117
- rba_string_homology_intra ....................................................... 118
- rba_string_interactions_network ............................................. 120
- rba_string_interaction_partners .............................................. 122
- rba_string_map_ids ............................................................... 124
- rba_string_network_image ....................................................... 126
- rba_string_version ............................................................... 129
- rba_uniprot_antigens ............................................................. 130
- rba_uniprot_antigens_search .................................................. 131
- rba_uniprot_coordinates ......................................................... 132
- rba_uniprot_coordinates_location ........................................... 134
- rba_uniprot_coordinates_sequence ......................................... 135
- rba_uniprot_coordinates_search ............................................. 137
- rba_uniprot_features ............................................................. 138
- rba_uniprot_features_search .................................................. 140
- rba_uniprot_gene-centric ........................................................ 142
- rba_uniprot_gene-centric_search .......................................... 143
- rba_uniprot_mutagenesis ......................................................... 145
- rba_uniprot_mutagenesis_search ............................................. 146
- rba_uniprot_proteins ............................................................. 147
- rba_uniprot_proteins_crossref .............................................. 149
- rba_uniprot_proteins_search .................................................. 150
- rba_uniprot_proteomes .......................................................... 153
rba_connection_test

Test if the Supported Services Are Responding

Description

Run this function to test the internet connectivity of your device and the current status of the supported Services.

Usage

rba_connection_test(print_output = TRUE, diagnostics = FALSE)

Arguments

print_output  (Logical) (default = TRUE) Send the tests’ output to the console?
diagnostics  (Logical) (default = FALSE) Show diagnostics and detailed messages with internal information.

Details

This function attempts to send a simple query to the supported services. If the service successfully responded, you will be informed with a success message; If not, the content of the error will be reported to you.
Please run this function if you encounter any errors while using rbioapi. Also, if you need to contact support, kindly call this function with ‘diagnostic = TRUE’ and include the output messages in your support request.
**Value**

Connection test for the supported servers will be displayed in console and the results will be invisibly returned as a list.

**See Also**

Other "Helper functions": `rba_options()`, `rba_pages()`

**Examples**

```r
rba_connection_test()
```

---

### Arguments

- **gene_list**
  - A vector with Entrez gene symbols.

- **description**
  - (optional) A name or description to be associated with your uploaded gene-set to Enrichr servers.

- **gene_set_library**
  - One of the:
    1. "all" to select all of the available Enrichr gene-set libraries.
    2. A gene-set library name existed in the results retrieved via `rba_enrichr_libs`
    3. If `regex_library_name` = TRUE, A partially-matching name a regex pattern that correspond to one or more of Enrichr library names.
regex_library_name
  logical: if TRUE (default) the supplied gene_set_library will be regarded as a
  regex or partially matching name. if FALSE, gene_set_library will be consid-
  ered exact match.

organism
  (default = "human") Which model organism version of Enrichr to use? Available
  options are: "human", (H. sapiens & M. musculus), "fly" (D. melanogaster),
  "yeast" (S. cerevisiae), "worm" (C. elegans) and "fish" (D. rerio).

progress_bar
  logical: In case of selecting multiple Enrichr libraries, should a progress bar be
  displayed?

... rbioapi option(s). See `rba_options`'s arguments manual for more information
  on available options.

Details
This function will call other rba_enrichr_*** functions with the following order:

  1. (If necessary) Call `rba_enrichr_libs` to obtain a list of available libraries in Enrichr.
  2. Call `rba_enrichr_add_list` to upload your gene-list and obtain a `user list ID`.
  3. Call `rba_enrichr_enrich` to perform enrichment analysis on the gene-list against one or
     multiple Enrichr libraries

Value
A list containing data frames of the enrichment results of your supplied gene-list against the selected
Enrichr libraries.

Corresponding API Resources
"GET https://maayanlab.cloud/Enrichr/datasetStatistics"
"POST https://maayanlab.cloud/Enrichr/addList"
"GET https://maayanlab.cloud/Enrichr/enrich"

References
• Chen, E.Y., Tan, C.M., Kou, Y. et al. Enrichr: interactive and collaborative HTML5 gene list
  14-128

• Maxim V. Kuleshov, Matthew R. Jones, Andrew D. Rouillard, Nicolas F. Fernandez, Qiao-
  nan Duan, Zichen Wang, Simon Koplev, Sherry L. Jenkins, Kathleen M. Jagodnik, Alexander
  Lachmann, Michael G. McDermott, Caroline D. Monteiro, Gregory W. Gundersen, Avi
  Ma’ayan, Enrichr: a comprehensive gene set enrichment analysis web server 2016 update, Nu-

• Xie, Z., Bailey, A., Kuleshov, M. V., Clarke, D. J. B., Evangelista, J. E., Jenkins, S. L., Lach-
  10.1002/cpz1.90

• Enrichr API Documentation

• Citations note on Enrichr website
See Also

Other "Enrichr": `rba_enrichr_add_list()`, `rba_enrichr_enrich()`, `rba_enrichr_gene_map()`, `rba_enrichr_libs()`, `rba_enrichr_view_list()`

Examples

```
## Not run:
rba_enrichr(gene_list = c("TP53", "TNF", "EGFR"))

## End(Not run)

rba_enrichr(gene_list = c("TP53", "TNF", "EGFR"),
             gene_set_library = "GO_Molecular_Function_2017",
             regex_library_name = FALSE)

rba_enrichr(gene_list = c("TP53", "TNF", "EGFR"),
             gene_set_library = "go",
             regex_library_name = TRUE)
```

---

**rba_enrichr_add_list**  *Upload Your Gene-List to Enrichr*

### Description

Prior to perform enrichment, Enrichr requires you to upload your gene-list and retrieve a 'user list ID'.

### Usage

```
rba_enrichr_add_list(gene_list, description = NULL, organism = "human", ...)
```

### Arguments

- **gene_list**
  - A vector with Entrez gene symbols.

- **description**
  - (optional) A name or description to be associated with your uploaded gene-set to Enrichr servers.

- **organism**
  - (default = "human") Which model organism version of Enrichr to use? Available options are: "human", (H. sapiens & M. musculus), "fly" (D. melanogaster), "yeast" (S. cerevisiae), "worm" (C. elegans) and "fish" (D. rerio).

- **...**
  - rbioapi option(s). See `rba_options`'s arguments manual for more information on available options.
Details

Note that using `rba_enrichr` is a more convenient way to automatically perform this and other required function calls to perform enrichment analysis on your input gene-set.

Value

A list with two unique IDs for your uploaded gene sets.

Corresponding API Resources

"POST https://maayanlab.cloud/Enrichr/addList"

References

- Enrichr API Documentation
- Citations note on Enrichr website

See Also

`rba_enrichr`

Other "Enrichr": `rba_enrichr()`, `rba_enrichr_enrich()`, `rba_enrichr_gene_map()`, `rba_enrichr_libs()`, `rba_enrichr_view_list()`

Examples

```r
rba_enrichr_add_list(gene_list = c("TP53", "TNF", "EGFR"),
                      description = "tumoral genes")
```
rba_enrichr_enrich  Get Enrichr Enrichment Results

Description

This function which will retrieve the enrichment results of your supplied gene-list id against one or multiple Enrichr libraries.

Usage

rba_enrichr_enrich(
  user_list_id,
  gene_set_library = "all",
  regex_library_name = TRUE,
  organism = "human",
  progress_bar = TRUE,
  ...
)

Arguments

user_list_id  An ID returned to you after uploading a gene list using rba_enrichr_add_list

gene_set_library

  One of the:

  1. "all" to select all of the available Enrichr gene-set libraries.
  2. A gene-set library name existed in the results retrieved via rba_enrichr_libs
  3. If regex_library_name = TRUE, A partially-matching name a regex pattern that correspond to one or more of Enrichr library names.

regex_library_name

  logical: if TRUE (default) the supplied gene_set_library will be regarded as a regex or partially matching name. if FALSE, gene_set_library will be considered exact match.

organism  (default = "human") Which model organism version of Enrichr to use? Available options are: "human", (H. sapiens & M. musculus), "fly" (D. melanogaster), "yeast" (S. cerevisiae), "worm" (C. elegans) and "fish" (D. rerio).

progress_bar  logical: In case of selecting multiple Enrichr libraries, should a progress bar be displayed?

...  rbioapi option(s). See rba_options's arguments manual for more information on available options.

Details

Note that using rba_enrichr is a more convenient way to automatically perform this and other required function calls to perform enrichment analysis on your input gene-set.
Value

A list containing data frames of the enrichment results of your supplied gene-list against the selected Enrichr libraries.

Corresponding API Resources

"GET https://maayanlab.cloud/Enrichr/enrich"

References

- Enrichr API Documentation
- Citations note on Enrichr website

See Also

rba_enrichr

Other "Enrichr": rba_enrichr(), rba_enrichr_add_list(), rba_enrichr_gene_map(), rba_enrichr_libs(), rba_enrichr_view_list()

Examples

```r
## Not run:
rba_enrichr_enrich(user_list_id = "11111")

## End(Not run)
## Not run:
rba_enrichr_enrich(user_list_id = "11111",
                   gene_set_library = "GO_Molecular_Function_2017",
                   regex_library_name = FALSE)

## End(Not run)
## Not run:
rba_enrichr_enrich(user_list_id = "11111",
                   gene_set_library = "go",
                   regex_library_name = TRUE)

## End(Not run)
```
rba_enrichr_gene_map  

Find Enrichr Terms That Contain a Given Gene

Description
This function will search the gene and retrieve a list of Enrichr Terms that contains that gene.

Usage
rba_enrichr_gene_map(gene, catagorize = FALSE, organism = "human", ...)

Arguments
- catagorize: logical: Should the category informations be included?
- organism: (default = "human") Which model organism version of Enrichr to use? Available options are: "human", (H. sapiens & M. musculus), "fly" (D. melanogaster), "yeast" (S. cerevisiae), "worm" (C. elegans) and "fish" (D. rerio).
- ... rbioapi option(s). See rba_options’s arguments manual for more information on available options.

Value
a list containing the search results of your supplied gene.

Corresponding API Resources
"GET https://maayanlab.cloud/Enrichr/genemap"

References
- Enrichr API Documentation
- Citations note on Enrichr website
See Also

Other "Enrichr": `rba_enrichr()`, `rba_enrichr_add_list()`, `rba_enrichr_enrich()`, `rba_enrichr_libs()`, `rba_enrichr_view_list()`

Examples

```r
rba_enrichr_gene_map(gene = "p53")

rba_enrichr_gene_map(gene = "p53", catagorize = TRUE)
```

---

```r
rba_enrichr_libs
```

### rba_enrichr_libs

**Retrieve a List of available libraries from Enrichr**

**Description**

This function will retrieve a list of available libraries in Enrichr with their statistics. And by default, will save those names as a global option ("rba_enrichr_libs") to be available for other Enrichr functions that internally require the names of Enrichr libraries.

**Usage**

```r
rba_enrichr_libs(store_in_options = FALSE, organism = "human", ...)
```

**Arguments**

- `store_in_options`  
  logical: (default = TRUE) Should a list of available Enrichr libraries be saved as a global option?

- `organism`  
  (default = "human") Which model organism version of Enrichr to use? Available options are: "human", (H. sapiens & M. musculus), "fly" (D. melanogaster), "yeast" (S. cerevisiae), "worm" (C. elegans) and "fish" (D. rerio).

- `...`  
  rbioapi option(s). See `rba_options`'s arguments manual for more information on available options.

**Details**

You should call this function once per R session with the argument 'store_in_options = TRUE' before using `rba_enrichr_enrich` or `rba_enrichr`. Nevertheless, rbioapi will do this for you in the background at the first time you call any function which requires this.

Note that using `rba_enrichr` is a more convenient way to automatically perform this and other required function calls to enrich your input gene-set.
Value

A data frame with the names of available library in Enrichr and their statistics.

Corresponding API Resources

"GET https://maayanlab.cloud/Enrichr/datasetStatistics"

References

- Enrichr API Documentation
- Citations note on Enrichr website

See Also

rba_enrichr

Other "Enrichr": rba_enrichr(), rba_enrichr_add_list(), rba_enrichr_enrich(), rba_enrichr_gene_map(), rba_enrichr_view_list()

Examples

rba_enrichr_libs()
Arguments

user_list_id    a user_list_id returned to you after uploading a gene list using `rba_enrichr_add_list`
organism        (default = "human") Which model organism version of Enrichr to use? Available options are: "human", (H. sapiens & M. musculus), "fly" (D. melanogaster), "yeast" (S. cerevisiae), "worm" (C. elegans) and "fish" (D. rerio).

...    rbioapi option(s). See `rba_options`’s arguments manual for more information on available options.

Value

A list containing the genes and description available under the supplied user_list_id

Corresponding API Resources

"GET https://maayanlab.cloud/Enrichr/view"

References

- Enrichr API Documentation
- Citations note on Enrichr website

See Also

Other "Enrichr": `rba_enrichr()`, `rba_enrichr_add_list()`, `rba_enrichr_enrich()`, `rba_enrichr_gene_map()`, `rba_enrichr.libs()`

Examples

```r
## Not run:
rba_enrichr_view_list(user_list_id = 11111)
```

```r
## End(Not run)```
rba_jaspar_collections

List collections available in JASPAR

Description

JASPAR organizes matrix profiles into collections. Using this function, you can retrieve a list of available collections in a JASPAR release.

Usage

rba_jaspar_collections(release = 2024, ...)

Arguments

...
rbioapi option(s). See rba_options’s arguments manual for more information on available options.

Value

A data frame with collections’ names and URLs.

Corresponding API Resources

"GET https://jaspar.elixir.no/api/v1/collections/

References

- JASPAR API Documentation
- Citations note on JASPAR website

See Also

Other "JASPAR": rba_jaspar_collections_matrices(), rba_jaspar_matrix(), rba_jaspar_matrix_search(), rba_jaspar_matrix_versions(), rba_jaspar_releases(), rba_jaspar_sites(), rba_jaspar_species(), rba_jaspar_species_matrices(), rba_jaspar_taxons(), rba_jaspar_taxons_matrices(), rba_jaspar_tffm(), rba_jaspar_tffm_search()
Examples

```r
rba_jaspar_collections(release = 2024)
```

---

List matrices available in a JASPAR collection

**Description**

Using this function you can list all matrix profiles that are available in a collection from a JASPAR release.

**Usage**

```r
rba_jaspar_collections_matrices(
  collection,
  release = 2024,
  only_last_version = FALSE,
  search = NULL,
  order = NULL,
  page_size = 1000,
  page = 1,
  ...
)
```

**Arguments**

- `collection` JASPAR Collection's name. See JASPAR Collections for information. The accepted values are: "CORE", "CNE", "PHYLOFACTS", "SPLICE", "POLII", "FAM", "PBM", "PBM_HOMEO", "PBM_HLH", and "UNVALIDATED".
- `only_last_version` Logical: (default = FALSE) If TRUE, only the latest version of a matrix profile will be returned.
- `search` Character: A search term.
- `order` Character: A character string or a vector of character strings of field names that will be used to order the results. Providing multiple field names is supported. You can also use prefix "." before a field name to indicate reverse ordering.
- `page_size` Numeric: (default = 1000) This resource returns paginated results. What is the maximum numbers of results that you want to retrieve per a page? Accepted values are between 1 and 1000.
Numeric: Which page of the results to retrieve? The accepted values depend on the page size and number of results.

rbaioapi option(s). See rba_options’s arguments manual for more information on available options.

Details

The results are paginated. You can control the page’s size number with the function’s arguments. Also, you can use rba_pages to automatically iterate over multiple pages.

Value

A list that contains a data frame with information of matrix profiles available in the collection.

Corresponding API Resources

"GET https://jaspar.elixir.no/api/v1/collections/{collection}/"

References


• JASPAR API Documentation

• Citations note on JASPAR website

See Also

Other "JASPAR": rba_jaspar_collections(), rba_jaspar_matrix(), rba_jaspar_matrix_search(),
  rba_jaspar_matrix_versions(), rba_jaspar_releases(), rba_jaspar_sites(), rba_jaspar_species(),
  rba_jaspar_species_matrices(), rba_jaspar_taxons(), rba_jaspar_taxons_matrices(),
  rba_jaspar_tffm(), rba_jaspar_tffm_search()

Examples

rba_jaspar_collections_matrices(collection = "CORE",
  release = 2024,
  page_size = 100,
  page = 2)
**rba_jaspar_matrix**  
Get a Position Frequency Matrices (PFM) with annotations

**Description**
Using this function you can retrieve a Position Frequency Matrices (PFM) associated with a matrix profile Identifier along with its details and annotations. If a base ID (i.e. without version suffix) was supplied, the latest version will be returned.

**Usage**
rba_jaspar_matrix(matrix_id, file_format = NULL, save_to = NULL, ...)

**Arguments**
- **matrix_id** Character: A matrix profile Identifier. It has "base_id.version" naming schema.
- **file_format** Character: Instead of returning a R object, you can directly download the profile matrix in file with this format. Supported formats are: "yaml", "jaspar", "transfac", "meme" and "pfm"
- **save_to** NULL or Character:
  - NULL: (only if file_format was supplied) Save the file to an automatically-generated path.
  - Character string: A valid file or directory path to save the file to.
- ... rbioapi option(s). See rba_options’s arguments manual for more information on available options.

**Value**
A list that contains the PFM along with its details and annotations. If file_format was supplied, an un-parsed character string with the file’s content.

**Corresponding API Resources**
"GET https://jaspar.elixir.no/api/v1/matrix/{matrix_id}/"

**References**
- JASPAR API Documentation
- Citations note on JASPAR website
See Also

Other "JASPAR": `rba_jaspar_collections()`, `rba_jaspar_collections_matrices()`, `rba_jaspar_matrix_search()`, `rba_jaspar_matrix_versions()`, `rba_jaspar_releases()`, `rba_jaspar_species()`, `rba_jaspar_species_matrices()`, `rba_jaspar_taxons()`, `rba_jaspar_taxons_matrices()`, `rba_jaspar_tffm()`, `rba_jaspar_tffm_search()`

Examples

```
rba_jaspar_matrix("MA0600.2")

## Not run:
rba_jaspar_matrix(matrix_id = "MA0600.2",
                   file_format = "meme",
                   save_to = "my_matrix.meme")

## End(Not run)
```

```
rba_jaspar_matrix_search

Search matrix profiles available in JASPAR

Description

You can use this function to list the JASPAR matrix profiles that match your search query, or run the function without any arguments to return a list of every matrix profile available in the latest release.

Usage

```
rba_jaspar_matrix_search(
    term = NULL,
    tf_name = NULL,
    tf_class = NULL,
    tf_family = NULL,
    tax_group = NULL,
    tax_id = NULL,
    data_type = NULL,
    collection = NULL,
    release = 2024,
    only_last_version = FALSE,
    order = NULL,
    page_size = 1000,
    page = 1,
    ...
)
```
```
Arguments

- **term**: Character: A search term.
- **tf_name**: Character: Transcription factor names (Case-sensitive).
- **tf_class**: Character: Transcription factor class
- **tf_family**: Character: Transcription factor family
- **tax_group**: Character: Taxonomic group. Use `rba_jaspar_taxons` to get a list of supported Taxonomic groups.
- **tax_id**: Numeric: NCBI taxonomic Identifier of species. Use `rba_jaspar_species` to get a list of supported Species.
- **data_type**: Character: Type of the data (i.e The Methodology used for matrix construction). For example: "ChIP-seq", "PBM"
- **collection**: Character: JASPAR matrix profile collection name. USE `rba_jaspar_collections` to get a list of collection names.
- **only_last_version**: Logical: (default = FALSE) If TRUE, only the latest version of a matrix profile will be returned.
- **order**: Character: A character string or a vector of character strings of field names that will be used to order the results. Providing multiple field names is supported. You can also use prefix ".-" before a field name to indicate reverse ordering.
- **page_size**: Numeric: (default = 1000) This resource returns paginated results. What is the maximum numbers of results that you want to retrieve per a page? Accepted values are between 1 and 1000.
- **page**: Numeric: Which page of the results to retrieve? The accepted values depend on the page size and number of results.
- ... rbioapi option(s). See `rba_options`’s arguments manual for more information on available options.

Details

Note that this is a search function. Thus, you are not required to fill every argument; You may use whatever combinations of arguments you see fit for your query. The results are paginated. You can control the page’s size number with the function’s arguments. Also, you can use `rba_pages` to automatically iterate over multiple pages.

Value

A list that contains a data frame of matrix profiles’ information.

Corresponding API Resources

"GET https://jaspar.elixir.no/api/v1/api/v1/matrix/"
**References**

- JASPAR API Documentation
- Citations note on JASPAR website

**See Also**

Other "JASPAR": `rba_jaspar_collections()`, `rba_jaspar_collections_matrices()`, `rba_jaspar_matrix()`, `rba_jaspar_matrix_versions()`, `rba_jaspar_releases()`, `rba_jaspar_sites()`, `rba_jaspar_species()`, `rba_jaspar_species_matrices()`, `rba_jaspar_taxons()`, `rba_jaspar_taxons_matrices()`, `rba_jaspar_tffm()`, `rba_jaspar_tffm_search()

**Examples**

```r
rba_jaspar_matrix_search(term = "FOX")
rba_jaspar_matrix_search(tf_name = "FOXP3")
rba_jaspar_matrix_search(tf_name = "FOXP3", only_last_version = TRUE)
rba_jaspar_matrix_search(tf_class = "Zipper-Type")
rba_jaspar_matrix_search(tax_group = "insects")
rba_jaspar_matrix_search(page_size = 100)
```

---

**rba_jaspar_matrix_versions**

*List matrix profile versions associated with a base ID*

**Description**

Since JASPAR release 2010, the matrix profiles are versioned; So, a matrix profile Identifier has "base_id.version" naming schema. Using this function you can retrieve a list of matrix profiles associated with a base (stable) ID.

**Usage**

```r
rba_jaspar_matrix_versions(base_id, order = NULL, ...)
```
Arguments

base_id  Character: A base (stable) Identifier. A matrix profile identifier has "base_id.version" naming schema

order  Character: A character string or a vector of character strings of field names that will be used to order the results. Providing multiple field names is supported. You can also use prefix ".-" before a field name to indicate reverse ordering.

...  rbioapi option(s). See `rba_options`'s arguments manual for more information on available options.

Value

A data frame of matrix profiles’ versions information.

Corresponding API Resources

"GET https://jaspar.elixir.no/api/v1/matrix/{base_id}/versions/"

References

- JASPAR API Documentation
- Citations note on JASPAR website

See Also

Other "JASPAR": `rba_jaspar_collections()`, `rba_jaspar_collections_matrices()`, `rba_jaspar_matrix()`, `rba_jaspar_matrix_search()`, `rba_jaspar_releases()`, `rba_jaspar_sites()`, `rba_jaspar_species()`, `rba_jaspar_species_matrices()`, `rba_jaspar_taxons()`, `rba_jaspar_taxons_matrices()`, `rba_jaspar_tffm()`, `rba_jaspar_tffm_search()`

Examples

```
rba_jaspar_matrix_versions("MA0600")
```
rba_jaspar_releases  Get information about JASPAR database releases

Description
If a release number was supplied, this function will return the details of that release. Otherwise, if
the function was called without "release" argument, a list of all JASPAR database releases will be
returned.

Usage
rba_jaspar_releases(release_number = NULL, ...)

Arguments
release_number Numeric: Which JASPAR database release number information’s to retrieve? If
left NULL (the default), a list of all JASPAR database releases will be returned.
Available options are 1 to 8.
... rbioapi option(s). See rba_options’s arguments manual for more information
on available options.

Value
A list that contains all JASPAR database releases’ information or details of a particular release.

Corresponding API Resources
"GET https://jaspar.elixir.no/api/v1/releases/
"GET https://jaspar.elixir.no/api/v1/releases/{release_number}/"

References
• Rauluseviciute I, Riudavets-Puig R, Blanc-Mathieu R, Castro-Mondragon JA, Ferenc K, Ku-
mar V, Lemma RB, Lucas J, Chèneby J, Baranasic D, Khan A, Fornes O, Gundersen S, Jo-
hansen M, Hovig E, Lenhard B, Sandelin A, Wasserman WW, Parcy F, Mathelier A JASPAR
2024: 20th anniversary of the open-access database of transcription factor binding profiles
Nucleic Acids Res. in_press; doi: 10.1093/nar/gkad1059
• Khan, A. and Mathelier, A. JASPAR RESTful API: accessing JASPAR data from any pro-
• JASPAR API Documentation
• Citations note on JASPAR website

See Also
Other "JASPAR": rba_jaspar_collections(), rba_jaspar_collections_matrices(), rba_jaspar_matrix(),
rba_jaspar_matrix_search(), rba_jaspar_matrix_versions(), rba_jaspar_sites(), rba_jaspar_species(),
rba_jaspar_species_matrices(), rba_jaspar_taxons(), rba_jaspar_taxons_matrices(),
rba_jaspar_tffm(), rba_jaspar_tffm_search()
Examples

rba_jaspar_releases()
rba_jaspar_releases(7)

---

**rba_jaspar_sites**  
*Get binding sites of a matrix profile*

Description

Use this function to retrieve a list of transcription factor binding sites associated with a matrix profile.

Usage

rba_jaspar_sites(matrix_id, ...)

Arguments

matrix_id  
Character: A matrix profile Identifier. It has "base_id.version" naming schema.

...  
rbiopsis option(s). See `rba_options`'s arguments manual for more information on available options.

Value

A list that contains a data frame with binding sites information.

Corresponding API Resources

"GET https://jaspar.elixir.no/api/v1/sites/{matrix_id}/"

References

- JASPAR API Documentation
- Citations note on JASPAR website
rba_jaspar_species

See Also

Other "JASPAR": rba_jaspar_collections(), rba_jaspar_collections_matrices(), rba_jaspar_matrix(), rba_jaspar_matrix_search(), rba_jaspar_matrix_versions(), rba_jaspar_releases(), rba_jaspar_species(), rba_jaspar_species_matrices(), rba_jaspar_taxons(), rba_jaspar_taxons_matrices(), rba_jaspar_tffm(), rba_jaspar_tffm_search()

Examples

rba_jaspar_sites("MA0600.1")

rba_jaspar_species List available species in JASPAR

Description

JASPAR organizes matrix profiles from multiple species in six taxonomic groups. Use this function to retrieve a list of available species in a JASPAR database release.

Usage

rba_jaspar_species(release = 2024, search = NULL, order = NULL, ...)

Arguments

search Character: A search term.
order Character: A character string or a vector of character strings of field names that will be used to order the results. Providing multiple field names is supported. You can also use prefix ".-" before a field name to indicate reverse ordering.
... rbioapi option(s). See rba_options’s arguments manual for more information on available options.

Value

A data frame with information of available species.

Corresponding API Resources

"GET https://jaspar.elixir.no/api/v1/species/"
References

- JASPAR API Documentation
- Citations note on JASPAR website

See Also

Other "JASPAR": `rba_jaspar_collections()`, `rba_jaspar_collections_matrices()`, `rba_jaspar_matrix()`, `rba_jaspar_matrix_search()`, `rba_jaspar_matrix_versions()`, `rba_jaspar_releases()`, `rba_jaspar_sites()`, `rba_jaspar_species_matrices()`, `rba_jaspar_taxons()`, `rba_jaspar_taxons_matrices()`, `rba_jaspar_tffm()`, `rba_jaspar_tffm_search()`

Examples

```r
rba_jaspar_species(release = 2024)
```

---

`rba_jaspar_species_matrices`

*List matrices available in JASPAR of a species*

Description

JASPAR curates matrix profiles from multiple species in six taxonomic groups. Using this function you can list all matrix profiles that are available in a JASPAR release from a species.

Usage

```r
rba_jaspar_species_matrices(
  tax_id,
  release = 2024,
  only_last_version = FALSE,
  search = NULL,
  order = NULL,
  page_size = 1000,
  page = 1,
  ...
)
```
Arguments

**tax_id**
Numeric: NCBI taxonomic Identifier of species. Use `rba_jaspar_species` to get a list of supported Species.

**release**

**only_last_version**
Logical: (default = FALSE) If TRUE, only the latest version of a matrix profile will be returned.

**search**
Character: A search term.

**order**
Character: A character string or a vector of character strings of field names that will be used to order the results. Providing multiple field names is supported. You can also use prefix "-" before a field name to indicate reverse ordering.

**page_size**
Numeric: (default = 1000) This resource returns paginated results. What is the maximum numbers of results that you want to retrieve per a page? Accepted values are between 1 and 1000.

**page**
Numeric: Which page of the results to retrieve? The accepted values depend on the page size and number of results.

... `rbioapi` option(s). See `rba_options`’s arguments manual for more information on available options.

Details

The results are paginated. You can control the page’s size number with the function’s arguments. Also, you can use `rba_pages` to automatically iterate over multiple pages.

Value

A list that contains a data frame with information of matrix profiles available for the species.

Corresponding API Resources

"GET https://jaspar.elixir.no/api/v1/species/\{tax_id\}/""

References

- JASPAR API Documentation
- Citations note on JASPAR website
See Also

Other "JASPAR": rba_jaspar_collections(), rba_jaspar_collections_matrices(), rba_jaspar_matrix(), rba_jaspar_matrix_search(), rba_jaspar_matrix_versions(), rba_jaspar_releases(), rba_jaspar_sites(), rba_jaspar_species(), rba_jaspar_taxons(), rba_jaspar_taxons_matrices(), rba_jaspar_tffm(), rba_jaspar_tffm_search()

Examples

rba_jaspar_species_matrices(tax_id = 9606, page_size = 100)

rba_jaspar_taxons(release = 2024, ...)

Arguments


Value

A data frame with information of available species.

Corresponding API Resources

"GET https://jaspar.elixir.no/api/v1/taxon"

References

rba_jaspar_taxons_matrices

- JASPAR API Documentation
- Citations note on JASPAR website

See Also

Other "JASPAR": rba_jaspar_collections(), rba_jaspar_collections_matrices(), rba_jaspar_matrix(), rba_jaspar_matrix_search(), rba_jaspar_matrix_versions(), rba_jaspar_releases(), rba_jaspar_sites(), rba_jaspar_species(), rba_jaspar_species_matrices(), rba_jaspar_taxons_matrices(), rba_jaspar_tffm(), rba_jaspar_tffm_search()

Examples

rba_jaspar_taxons(release = 2024)

rba_jaspar_taxons_matrices

List matrices available in JASPAR of a taxonomic group

Description

JASPAR organizes matrix profiles from multiple species in six taxonomic groups. Using this function you can list all matrix profiles that are available in a JASPAR release from a taxonomic group.

Usage

rba_jaspar_taxons_matrices(
  tax_group,
  release = 2024,
  only_last_version = FALSE,
  search = NULL,
  order = NULL,
  page_size = 1000,
  page = 1,
  ...
)

Arguments

tax_group Character: Taxonomic group. Use rba_jaspar_taxons to get a list of supported Taxonomic groups.
only_last_version
Logical: (default = FALSE) If TRUE, only the latest version of a matrix profile will be returned.

search
Character: A search term.

order
Character: A character string or a vector of character strings of field names that will be used to order the results.
Providing multiple field names is supported. You can also use prefix "-" before a field name to indicate reverse ordering.

page_size
Numeric: (default = 1000) This resource returns paginated results. What is the maximum numbers of results that you want to retrieve per a page? Accepted values are between 1 and 1000.

page
Numeric: Which page of the results to retrieve? The accepted values depend on the page size and number of results.

... rbioapi option(s). See rba_options’s arguments manual for more information on available options.

Details
The results are paginated. You can control the page’s size number with the function’s arguments. Also, you can use rba_pages to automatically iterate over multiple pages.

Value
A list that contains a data frame with information of matrix profiles available for the taxonomic group.

Corresponding API Resources
"GET https://jaspar.elixir.no/api/v1/taxon/{tax_group}/"

References
- JASPAR API Documentation
- Citations note on JASPAR website

See Also
Other "JASPAR": rba_jaspar_collections(), rba_jaspar_collections_matrices(), rba_jaspar_matrix(), rba_jaspar_matrix_search(), rba_jaspar_matrix_versions(), rba_jaspar_releases(), rba_jaspar_sites(), rba_jaspar_species(), rba_jaspar_species_matrices(), rba_jaspar_taxons(), rba_jaspar_tffm(), rba_jaspar_tffm_search()
**rba_jaspar_tffm**

**Examples**

```r
rba_jaspar_taxons_matrices(tax_group = "plants", page_size = 100)
```

---

**rba_jaspar_tffm**  
*Get a TF flexible models (TFFMs) information*

**Description**

Using this function you can retrieve details and annotations of Transcription Factor flexible models (TFFMs) associated with a TFFM ID. If a base ID (i.e. without version suffix) was supplied, the latest version will be returned.

**Usage**

```r
rba_jaspar_tffm(tffm_id, ...)
```

**Arguments**

- **tffm_id**  
  Character: A TF flexible model (TFFM) Identifier.

- **...**  
  rbioapi option(s). See `rba_options`’s arguments manual for more information on available options.

**Value**

A list that contains the TFFM’s information and annotations.

**Corresponding API Resources**

"GET https://jaspar.elixir.no/api/v1/fttm/{tffm_id}/"

**References**

- JASPAR API Documentation
- Citations note on JASPAR website
See Also

Other "JASPAR": `rba_jaspar_collections()`, `rba_jaspar_collections_matrices()`, `rba_jaspar_matrix()`, `rba_jaspar_matrix_search()`, `rba_jaspar_matrix_versions()`, `rba_jaspar_releases()`, `rba_jaspar_sites()`, `rba_jaspar_species()`, `rba_jaspar_species_matrices()`, `rba_jaspar_taxons()`, `rba_jaspar_taxons_matrices()`, `rba_jaspar_tffm_search()`

Examples

```r
rba_jaspar_tffm("TFFM0056.3")
```

Description

You can use this function to list the JASPAR TF flexible models (TFFMs) that match your search query, or run the function without any arguments to return a list of every matrix profile available in the latest release.

Usage

```r
rba_jaspar_tffm_search(
  term = NULL,
  release = 2024,
  tax_group = NULL,
  search = NULL,
  order = NULL,
  page_size = 1000,
  page = 1,
  ...
)
```

Arguments

- **term** Character: A search term.
- **tax_group** Character: Taxonomic group. Use `rba_jaspar_taxons` to get a list of supported Taxonomic groups.
- **search** Character: A search term.
Character: A character string or a vector of character strings of field names that will be used to order the results. Providing multiple field names is supported. You can also use prefix "-" before a field name to indicate reverse ordering.

`page_size` Numeric: (default = 1000) This resource returns paginated results. What is the maximum numbers of results that you want to retrieve per a page? Accepted values are between 1 and 1000.

`page` Numeric: Which page of the results to retrieve? The accepted values depend on the page size and number of results.

... rbioapi option(s). See `rba_options`’s arguments manual for more information on available options.

Details

Note that this is a search function. Thus, you are not required to fill every argument; You may use whatever combinations of arguments you see fit for your query. The results are paginated. You can control the page’s size number with the function’s arguments. Also, you can use `rba_pages` to automatically iterate over multiple pages.

Value

A list that contains a data frame with information of query hits’ TFFMs.

Corresponding API Resources

"GET https://jaspar.elixir.no/api/v1/api/v1/tffm/

References

- JASPAR API Documentation
- Citations note on JASPAR website

See Also

Other "JASPAR": `rba_jaspar_collections()`, `rba_jaspar_collections_matrices()`, `rba_jaspar_matrix()`, `rba_jaspar_matrix_search()`, `rba_jaspar_matrix_versions()`, `rba_jaspar_releases()`, `rba_jaspar_sites()`, `rba_jaspar_species()`, `rba_jaspar_species_matrices()`, `rba_jaspar_taxons()`, `rba_jaspar_taxons_matrices()`, `rba_jaspar_tffm()`
Examples

```r
rba_jaspar_tffm_search(term = "FOX")
rba_jaspar_tffm_search(tax_group = "insects")
rba_jaspar_tffm_search(page_size = 100)
```

### rba_mieaa_cats

**Get Supported Enrichment Categories for a Species and miRNA Type**

**Description**

For each Combination of species and miRNA type, Only a pre-defined categories groups are supported. Use this function to retrieve a list of supported categories for a given combination of Species and miRNA type.

**Usage**

```r
rba_mieaa_cats(mirna_type, species, ...)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mirna_type</td>
<td>Type of your miRNA accession. either &quot;mature&quot; or &quot;precursor&quot;.</td>
</tr>
<tr>
<td>species</td>
<td>Fully or partially matching Scientific name, abbreviation or NCBI taxon ID of one of the following species:</td>
</tr>
<tr>
<td></td>
<td>1. &quot;Homo sapiens&quot;, &quot;hsa&quot; or 9606</td>
</tr>
<tr>
<td></td>
<td>2. &quot;Mus musculus&quot;, &quot;mmu&quot; or 10090</td>
</tr>
<tr>
<td></td>
<td>3. &quot;Rattus norvegicus&quot;, &quot;rno&quot; or 10116</td>
</tr>
<tr>
<td></td>
<td>4. &quot;Arabidopsis thaliana&quot;, &quot;ath&quot; or 3702</td>
</tr>
<tr>
<td></td>
<td>5. &quot;Bos taurus&quot;, &quot;bta&quot; or 9913</td>
</tr>
<tr>
<td></td>
<td>6. &quot;Caenorhabditis elegans&quot;, &quot;cel&quot; or 6239</td>
</tr>
<tr>
<td></td>
<td>7. &quot;Drosophila melanogaster&quot;, &quot;dme&quot; or 7227</td>
</tr>
<tr>
<td></td>
<td>8. &quot;Danio rerio&quot;, &quot;dre&quot; or 7955</td>
</tr>
<tr>
<td></td>
<td>9. &quot;Gallus gallus&quot;, &quot;gga&quot; or 9031</td>
</tr>
<tr>
<td></td>
<td>10. &quot;Sus scrofa&quot;, &quot;ssc&quot; or 9823</td>
</tr>
<tr>
<td></td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>rbioapi option(s). See <code>rba_options</code>'s arguments manual for more information on available options.</td>
</tr>
</tbody>
</table>

**Value**

a named character vector with the supported categories for your supplied input combination.

**Corresponding API Resources**

"GET https://ccb-compute2.cs.uni-saarland.de/mieaa2/api/v1/enrichment_categories/{species}/{mirna_type}="/
References

- miEAA browsable API tutorial
- Citations note on miEAA website

See Also

Other "miEAA": `rba_mieaa_convert_type()`, `rba_mieaa_convert_version()`, `rba_mieaa_enrich()`, `rba_mieaa_enrich_results()`, `rba_mieaa_enrich_status()`, `rba_mieaa_enrich_submit()`

Examples

```r
rba_mieaa_cats("mature", "Homo sapiens")
```

---

`rba_mieaa_convert_type`

Convert Between Mature and precursor miRNA Accession

Description

miRBase miRNA accession could refer to either mature or precursor miRNAs. (see: A uniform system for microRNA annotation). Use this function to mature miRNA accession to corresponding miRNA accessions or vice versa.

Usage

```r
rba_mieaa_convert_type(
  mirna,  # A vector of miRNA accessions to be converted.
  input_type,    # Type of your supplied miRNA accession. either "mature" or "precursor".
  only_unique = FALSE,
  simple_output = FALSE,
  ...
)
```

Arguments

- `mirna`  A vector of miRNA accessions to be converted.
- `input_type`  Type of your supplied miRNA accession. either "mature" or "precursor".
rba_mieaa_convert_type

only_unique (logical) miRBase precursor and mature miRNA accessions are not uniquely mapped. (i.e. you may get more than one results for a given accession). set this to TRUE to only retrieve the unique mappings. (default = FALSE)

simple_output (logical) If FALSE (default), the result will be a two-columned data frame with your input and output accessions. Otherwise, if TRUE, only the output miRNA accessions will be returned.

... rbioapi option(s). See rba_options’s arguments manual for more information on available options.

Value

Depending on the arguments, a data frame or a character vectors containing the miRNA accessions in your output version.

Corresponding API Resources

"POST https://ccb-compute2.cs.uni-saarland.de/mieaa2/api/v1/mirna_precursor_converter/"

References

• miEAA browsable API tutorial
• Citations note on miEAA website

See Also

Other "miEAA": rba_mieaa_cats(), rba_mieaa_convert_version(), rba_mieaa_enrich(), rba_mieaa_enrich_results(), rba_mieaa_enrich_status(), rba_mieaa_enrich_submit()

Examples

Sys.sleep(1) # to prevent 429 error during R CMD check
rba_mieaa_convert_type(mirna = c("hsa-miR-20b-5p", "hsa-miR-144-5p"),
input_type = "mature")
**rba_mieaa_convert_version**

*Convert miRNA accession Between Different miRBase Versions*

**Description**

miEAA works with miRBASE v22 accession. Using This function you can convert a set of mature or precursor miRNA accession between two given miRBase versions.

**Usage**

```r
rba_mieaa_convert_version(
  mirna,  # A vector of miRNA accessions to be converted.
  mirna_type,  # Type of your supplied miRNA accession. either "mature" or "precursor".
  input_version,  # (numeric) miRBase version of your supplied miRNA accessions.
  output_version,  # (numeric) To what version should your miRNA accessions be converted?
  simple_output = FALSE,  # (logical) If FALSE (default), the result will be a two-columned data frame with your input and output accessions. Otherwise, if TRUE, only the output miRNA accessions will be returned.
  ...  # rbioapi option(s). See rba_options’s arguments manual for more information on available options.
)
```

**Arguments**

- **mirna**: A vector of miRNA accessions to be converted.
- **mirna_type**: Type of your supplied miRNA accession. either "mature" or "precursor".
- **input_version**: (numeric) miRBase version of your supplied miRNA accessions.
- **output_version**: (numeric) To what version should your miRNA accessions be converted?
- **simple_output**: (logical) If FALSE (default), the result will be a two-columned data frame with your input and output accessions. Otherwise, if TRUE, only the output miRNA accessions will be returned.
- **...**: rbioapi option(s). See rba_options’s arguments manual for more information on available options.

**Value**

Depending on the arguments, a data frame or a character vectors containing the miRNA accessions in your output version.

**Corresponding API Resources**

"POST https://ccb-compute2.cs.uni-saarland.de/mieaa2/api/v1/mirbase_converter/"
References

- miEAA browsable API tutorial
- Citations note on miEAA website

See Also

Other "miEAA": rba_mieaa_cats(), rba_mieaa_convert_type(), rba_mieaa_enrich(), rba_mieaa_enrich_results(), rba_mieaa_enrich_status(), rba_mieaa_enrich_submit()

Examples

Sys.sleep(1) # to prevent 429 error during R CMD check
rba_mieaa_convert_version(mirna = c("hsa-miR-20b-5p", "hsa-miR-144-5p"),
                         mirna_type = "mature", input_version = 22, output_version = 16)

---

rba_mieaa_enrich A One-step Wrapper for miRNA Enrichment Using miEAA

Description

This function is a wrapper for the multiple function calls necessary to perform enrichment analysis on a given miRNA list using miEAA. see details section for more information.

Usage

rba_mieaa_enrich(
  test_set,
  mirna_type,
  test_type,
  species,
  categories = NULL,
  p_adj_method = "fdr",
  independent_p_adj = TRUE,
  sig_level = 0.05,
  min_hits = 2,
  ref_set = NULL,
  sort_by = "p_adjusted",
  sort_asc = TRUE,
  ...
)


Arguments

test_set a character vector with your mature or precursor miRBase miRNA accessions. Note that
1. Only miRBase v22 miRNA accession are accepted. You can use \texttt{rba_mieaa_convert_version}
to convert your accessions to miRBase v22.
2. Your list should be entirely consisted of either mature or precursor miRNA accession. A mixture of both is not accepted.

mirna_type Type of your supplied miRNA accession. either "mature" or "precursor".

mirna_type test_type The analysis to perform. can be either "ORA" for 'Over Representation Analysis' or "GSEA" for miRNA (Gene) 'Set Enrichment Analysis'. Note that in GSEA, your list should be sorted beforehand based on some criterion.

species Fully or partially matching Scientific name, abbreviation or NCBI taxon ID of one of the following species:
1. "Homo sapiens", "hsa" or 9606
2. "Mus musculus", "mmu" or 10090
3. "Rattus norvegicus", "rno" or 10116
4. "Arabidopsis thaliana", "ath" or 3702
5. "Bos taurus", "bta" or 9913
6. "Caenorhabditis elegans", "cel" or 6239
7. "Drosophila melanogaster", "dme" or 7227
8. "Danio rerio", "dre" or 7955
9. "Gallus gallus", "gga" or 9031
10. "Sus scrofa", "ssc" or 9823

categories one or multiple Category names to be used for miRNA set enrichment analysis. Note that
- Available categories varies based on your chosen specie and if your supplied miRNA type is mature or precursor. Use \texttt{rba_mieaa_cats} to retrieve a list of available category names for a given specie and miRNA type.
- If you supply NULL, the analysis will be performed on all of the available categories.

p_adj_method P-value adjustment method to be used. Should be one of: "none", "fdr" (default), "bonferroni", "BY", "hochberg", "holm" or "hommel"

independent_p_adj (logical) The scope and level of p-value adjustment; if TRUE (default), the categories will be considered independent from each other and the p-value will be adjusted separately for each category. if FALSE, the p-value will be adjusted collectively over all categories.

sig_level (numeric) The significance threshold of adjusted P-value. values equal to or greater than this threshold will be dropped from the results.

min_hits (numeric) How many miRNA should a sub-category have from your supplied test-list to be included in the results? (default is 2)

ref_set (Optional) Only applicable when test_type is "ORA". This character vector will be used as your reference (background or universe) set for p-value calculations.
sort_by A column name to the result’s table based on that. one of: "category", "subcategory", "enrichment", "p_value", "p_adjusted" (default), "q_value" or "observed".

sort_asc (logical) If TRUE, the results will be sorted in ascending order. If FALSE, the results will be sorted in descending order.

Details

This function will call other rba_mieaa_*** functions with the following order:

1. Call rba_mieaa_enrich_submit to Submit an enrichment analysis request to miEAA servers, using your supplied miRNA lists and other arguments.
2. Once your job was successfully submitted, it will call rba_mieaa_enrich_status every 5 seconds, to check the status of your running server-side job and whether your analysis job is finished and the results are available.
3. Call rba_mieaa_enrich_results to retrieve the results of your enrichment analysis.

See each function’s manual for more details.

Value

A data frame with your enrichment analysis results.

Corresponding API Resources

"GET https://ccb-compute2.cs.uni-saarland.de/mieaa2/api"

References

- miEAA browsable API tutorial
- Citations note on miEAA website

See Also

Other "miEAA": rba_mieaa_cats(), rba_mieaa_convert_type(), rba_mieaa_convert_version(), rba_mieaa_enrich_results(), rba_mieaa_enrich_status(), rba_mieaa_enrich_submit()
Examples

```r
## Not run:
rba_mieaa_enrich(test_set = c("hsa-miR-20b-5p", "hsa-miR-144-5p", "hsa-miR-17-5p", "hsa-miR-20a-5p"), mirna_type = "mature", test_type = "ORA", species = 9606, categories = "miRPathDB_GO_Biological_process_mature")

## End(Not run)
```

---

**rba_mieaa_enrich_results**

*Retrieve Results of a finished Enrichment Analysis from miEAA*

Description

After your submitted enrichment analysis request has finished (check using `rba_mieaa_enrich_status`), you can retrieve the results using this function.

Usage

```r
rba_mieaa_enrich_results(job_id, sort_by = "p_adjusted", sort_asc = TRUE, ...)
```

Arguments

- **job_id**
  - The job-id (a character string) of a submitted enrichment analysis.
- **sort_by**
  - A column name to the result’s table based on that. one of: "category", "subcategory", "enrichment", "p_value", "p_adjusted" (default), "q_value" or "observed".
- **sort_asc**
  - (logical) If TRUE, the results will be sorted in ascending order. If FALSE, the results will be sorted in descending order.
- **...**
  - rbioapi option(s). See `rba_options`'s arguments manual for more information on available options.

Details

Note that using `rba_mieaa_enrich` is a more convenient way to automatically perform this and other required function calls to perform enrichment analysis on your input miRNA-set using miEAA.

Value

A data frame with your enrichment analysis results.

Corresponding API Resources

"GET https://ccb-compute2.cs.uni-saarland.de/mieaa2/api/v1/results/{job_id}"

References

- miEAA browsable API tutorial
- Citations note on miEAA website

See Also

Other "miEAA": `rba_mieaa_cats()`, `rba_mieaa_convert_type()`, `rba_mieaa_convert_version()`, `rba_mieaa_enrich()`, `rba_mieaa_enrich_status()`, `rba_mieaa_enrich_submit()`

Examples

```r
## Not run:
rba_mieaa_enrich_results("f52d1aef-6d3d-4d51-9020-82e68fe99012")
## End(Not run)
```

```
rba_mieaa_enrich_status

Check Status of a Submitted Enrichment Analysis in miEAA

Description

After you have submitted your enrichment analysis (using `rba_mieaa_enrich_submit`) and retrieved a job-id, you can use this function to check the status of your job. Status value equal to 100 means that your requested analysis has finished and you may retrieve the results using `rba_mieaa_enrich_results`.

Usage

`rba_mieaa_enrich_status(job_id, ...)`

Arguments

- `job_id` The job-id (a character string) of a submitted enrichment analysis.
- `...` rbioapi option(s). See `rba_options`’s arguments manual for more information on available options.

Details

Note that using `rba_mieaa_enrich` is a more convenient way to automatically perform this and other required function calls to perform enrichment analysis on your input miRNA-set using miEAA.
Value

A list containing the status value for a analysis that corresponds to your supplied job-id.

Corresponding API Resources

"GET https://ccb-compute2.cs.uni-saarland.de/mieaa2/api/v1/job_status/{job_id}" 

References

• miEAA browsable API tutorial
• Citations note on miEAA website

See Also

Other "miEAA": rba_mieaa_cats(), rba_mieaa_convert_type(), rba_mieaa_convert_version(), rba_mieaa_enrich(), rba_mieaa_enrich_results(), rba_mieaa_enrich_submit()

Examples

## Not run:
Sys.sleep(1) # to prevent 429 error during R CMD check
rba_mieaa_enrich_status("f52d1aeef-6d3d-4d51-9020-82e68fe99012")

## End(Not run)

---

rba_mieaa_enrich_submit

Submit miEAA miRNA Enrichment Analysis Request

Description

Using This function you can submit a request in miEAA servers to perform Over-representation or GSEA Analysis for a given set of miRNA identifiers. see "arguments" section for more information.

Usage

rba_mieaa_enrich_submit(
  test_set,
  mirna_type,
  test_type,
  species = "hsa",
)
categories = NULL,
p_adj_method = "fdr",
independent_p_adj = TRUE,
sig_level = 0.05,
min_hits = 2,
ref_set = NULL,
...)

Arguments

test_set       a character vector with your mature or precursor miRBase miRNA accessions. Note that
               1. Only miRBase v22 miRNA accession are accepted. You can use `rba_mieaa_convert_version`
               to convert your accessions to miRBase v22.
               2. Your list should be entirely consisted of either mature or precursor miRNA
                  accession. A mixture of both is not accepted.

mirna_type     Type of your supplied miRNA accession. either "mature" or "precursor".

test_type      The analysis to perform. can be either "ORA" for 'Over Representation Analysis' or "GSEA" for miRNA (Gene) 'Set Enrichment Analysis'. Note that in GSEA, your list should be sorted beforehand based on some criterion.

species        Fully or partially matching Scientific name, abbreviation or NCBI taxon ID of one of the following species:
               1. "Homo sapiens", "hsa" or 9606
               2. "Mus musculus", "mmu" or 10090
               3. "Rattus norvegicus", "rno" or 10116
               4. "Arabidopsis thaliana", "ath" or 3702
               5. "Bos taurus", "bta" or 9913
               6. "Caenorhabditis elegans", "cel" or 6239
               7. "Drosophila melanogaster", "dme" or 7227
               8. "Danio rerio", "dre" or 7955
               9. "Gallus gallus", "gga" or 9031
               10. "Sus scrofa", "ssc" or 9823

categories     one or multiple Category names to be used for miRNA set enrichment analysis. Note that
               • Available categories varies based on your chosen specie and if your supplied miRNA type is mature or precursor. Use `rba_mieaa_cats` to retrieve a list of available category names for a given specie and miRNA type.
               • If you supply NULL, the analysis will be performed on all of the available categories.

p_adj_method   P-value adjustment method to be used. Should be one of: "none", "fdr" (default),
               "bonferroni", "BY", "hochberg", "holm" or "hommel"

independent_p_adj (logical) The scope and level of p-value adjustment; if TRUE (default), the categories will be considered independent from each other and the p-value will be
adjusted separately for each category. if FALSE, the p-value will be adjusted collectively over all categories.

*sig_level* (numeric) The significance threshold of adjusted P-value. values equal to or greater than this threshold will be dropped from the results.

*min_hits* (numeric) How many miRNA should a sub-category have from your supplied test-list to be included in the results? (default is 2)

*ref_set* (Optional) Only applicable when test_type is "ORA". This character vector will be used as your reference (background or universe) set for p-value calculations.

... rbioapi option(s). See rba_options’s arguments manual for more information on available options.

**Details**

Note that using rba_mieaa_enrich is a more convenient way to automatically perform this and other required function calls to perform enrichment analysis on your input miRNA-set using miEAA.

**Value**

A list that contains your submitted job’s ID and a URL to manually check for your job status.

**Corresponding API Resources**

"POST https://ccb-compute2.cs.uni-saarland.de/mieaa2/api/v1/enrichment_analysis/{species}/{type}/{test}/"

**References**

- miEAA browsable API tutorial
- Citations note on miEAA website

**See Also**

Other "miEAA": rba_mieaa_cats(), rba_mieaa_convert_type(), rba_mieaa_convert_version(), rba_mieaa_enrich(), rba_mieaa_enrich_results(), rba_mieaa_enrich_status()

**Examples**

```r
Sys.sleep(1) # to prevent 429 error during R CMD check
rba_mieaa_enrich_submit(test_set = c("hsa-miR-20b-5p", "hsa-miR-144-5p"),
mirna_type = "mature",
test_type = "GSEA",
species = 9606,
categories = NULL)
```
Set rbioapi Global Options

Description

A safe way to change rbioapi’s global options and behavior. See "arguments" section for available options.

Note that you are not limited to changing the options globally, you can include the option names and values in the '...' argument of any rbioapi function to alter the option(s) only in that function call; e.g. example_function(x, diagnostics = TRUE, timeout = 300).

Alternatively, you can call this function with no arguments, i.e. rba_options(), to retrieve a data frame of available rbioapi options and their current values.

Usage

rba_options(
  diagnostics = NULL,
  dir_name = NULL,
  retry_max = NULL,
  retry_wait = NULL,
  progress = NULL,
  save_file = NULL,
  skip_error = NULL,
  timeout = NULL,
  verbose = NULL
)

Arguments

- diagnostics
  - Logical (default = FALSE) Show diagnostics and detailed messages with internal information.

- dir_name
  - Character (default = "rbioapi") If the package needs to generate a file path to save the server’s response, a directory with this name will be created in your working directory to save your files.

- retry_max
  - Numeric (default = 0) How many times should rbioapi retry in case of 5xx server responses, errors related to the server or no internet connectivity?

- retry_wait
  - Numeric (default = 10) Time in seconds to wait before next retry in case of internet connection or server problems.

- progress
  - Logical (default = FALSE) Should a progress bar be displayed?

- save_file
  - Logical or character (default = FALSE) Either:
    - TRUE: In this case, the raw server’s response file will be automatically saved to a proper file path. Use "dir_name" argument to change the file’s parent directory.
    - FALSE: (default) Do not automatically save server’s response file.
rba_options

- Character: (Only when changing the option via "..." in a functions call) A valid file path to save the server's response file to the function that you are calling.

skip_error (Logical) (default = FALSE if R is in the interactive mode, TRUE otherwise) If TRUE, the code execution will not be stopped in case of errors (anything but HTTP status 200 from the server); Instead the error message will be returned as the function's output. However, if FALSE, in case of any error, the code execution will be halted and an error message will be issued.

timeout (Numeric) (default = 30) The maximum time in seconds that you are willing to wait for a server response before giving up and stopping the function execution.

verbose (Logical) (Default = TRUE) Generate short informative messages.

Details

Because this function validates your supplied changes, please only change rbioapi options using this function and avoid directly editing them.

Value

If called without any argument, a Data frame with available options and their information; If Called with an argument, will Return NULL but Alters that option globally.

See Also

Other "Helper functions": rba_connection_test(), rba_pages()

Examples

rba_options()
## Not run:
rba_options(verbos e = FALSE)

## End(Not run)
## Not run:
rba_options(save_file = TRUE)

## End(Not run)
## Not run:
rba_options(diagnostics = TRUE, progress = TRUE)

## End(Not run)
rba_pages

Get Multiple Pages of a Paginated Resource

Description

Some resources return paginated results, meaning that you have to make separate calls for each page. Using this function, you can iterate over up to 100 pages. Just supply your rbioapi function and change to page argument to "pages:start_page:end_page", for example "pages:1:5".

Usage

rba_pages(input_call, ...)

Arguments

input_call

A quoted call. supply a regular rbioapi function call, but with two differences:
1. Wrap a quote() around it. meaning: quote(rba_example())
2. Set the argument that corresponds to the page number to "pages:start_page:end_page", for example "pages:1:5".

See the "examples" section to learn more.

Details

To prevent flooding the server, there will be a 1 second delay between calls, also the user cannot iterate on more than 100 pages. The function will also override skip_error option and will always set it to TRUE. This means that in case of server response error (e.g. requesting pages that do not exist) an error message be returned to you instead of halting function’s execution.

Value

A named list where each element corresponds to a request’s page.

See Also

Other "Helper functions": rba_connection_test(), rba_options()

Examples

```r
rba_pages(input_call = quote(rba_uniprot_taxonomy(ids = 189831, hierarchy = "siblings", page_size = 50, page_number = "pages:1:5")))
```

```r
rba_pages(input_call = quote(rba_uniprot_taxonomy_name(name = "adenovirus",
```
**rba_panther_enrich**

**PANTHER Over-Representation Enrichment Analysis**

**Description**

Using this function you can use PANTHER services to perform over-representation enrichment analysis. This statistical test will compare your input genes to a set of defined gene lists to determine if they are over/under-represented.

**Usage**

```r
rba_panther_enrich(
  genes,
  organism,
  annot_dataset,
  test_type = "FISHER",
  correction = "FDR",
  cutoff = NULL,
  ref_genes = NULL,
  ref_organism = NULL,
  ...
)
```

**Arguments**

- **genes**: Character vector of genes identifiers with maximum length of 10000. Can be any of: Ensemble gene ID, Ensemble protein ID, Ensemble transcript ID, Entrez gene ID, gene symbol, NCBI GI, HGNC ID, International protein index ID, NCBI UniGene ID, UniProt accession and/or UniProt ID.
- **organism**: (numeric) NCBI taxon ID. run `rba_panther_info` with argument 'what = "organisms"' to get a list of PANTHER’s supported organisms.
- **annot_dataset**: A PANTHER dataset ID to test your input against it. run `rba_panther_info` with argument 'what = "datasets"' to get a list of PANTHER’s supported datasets. Note that you should enter the "id" of the dataset, not its label (e.g. entering "biological_process" is incorrect, you should rather enter "GO:0008150").
test_type  statistical test type to calculate the p values. either "FISHER" (default) or "BINO
MIAL".
correction p value correction method. either "FDR" (default), "BONFERRONI" or "NONE".
cutoff (Numeric) (Optional) a threshold to filter the results. if correction is "FDR", the
threshold will be applied to fdr column’s values; if otherwise, the threshold will be applied to p value column.
ref_genes (Optional) A set of genes that will be used as the test’s background (refer-
cence/universe) gene set. If no value supplied, all of the genes in specified organ-
ism will be used. maximum length and supported IDs are the same as ‘genes’
argument.
ref_organism (Optional) if ‘ref_genes’ is used, you can specify the organisms which corre-
spond to your supplied IDs in ‘ref_genes’ argument. see ‘organism’ argument
for supported values.

... rbioapi option(s). See rba_options’s arguments manual for more information
on available options.

Value
A list with the parameters and results. If the analysis was successful, the results data frame are
returned in the "results" element within the list. Otherwise, an error message will be returned under
the "search$error" element in the returned list.

Corresponding API Resources
"POST https://www.pantherdb.org/services/oai/pantherdb/enrich/overrep"

References
• Huaiyu Mi, Dustin Ebert, Anushya Muruganujan, Caitlin Mills, Laurent-Philippe Albou,
Tremayne Mushayamaha, Paul D Thomas, PANTHER version 16: a revised family classi-
fication, tree-based classification tool, enhancer regions and extensive API, Nucleic Acids Re-
search, Volume 49, Issue D1, 8 January 2021, Pages D394–D403, https://doi.org/10.1093/nar/gkaa1106
• PANTHER Services Details
• Citations note on PANTHER website

See Also
Other "PANTHER": rba_panther_family(), rba_panther_homolog(), rba_panther_info(),
rba_panther_mapping(), rba_panther_ortholog(), rba_panther_tree_grafter()

Examples
rba_panther_enrich(genes = c("p53", "BRCA1", "cdk2", "Q99835", "CDC42",
"CDK1", "KIF23", "PLK1", "RAC2", "RACGAP1"),
organism = 9606, annot_dataset = "GO:0008150",
cutoff = 0.01)
Description

Using this function, you can retrieve Orthologs, MSA or Tree topology information of a given PANTHER family.

Usage

rba_panther_family(id, what, target_organisms = NULL, ...)

Arguments

id
Panther family id.

what
What to retrieve? One of:

- "ortholog": Orthologs (‘LDO’ for least diverged and ‘O’ for more diverged).
- "msa": Multiple Sequence Alignment Information,
- "tree": Tree topology and nodes attributes.

target_organisms
(numeric) NCBI taxon ID(s) to filter the results. run rba_panther_info with argument ‘what = "organisms"’ to get a list of PANTHER’s supported organisms.

... rbioapi option(s). See rba_options’s arguments manual for more information on available options.

Value

For trees a list and otherwise a data frame with the requested family’s information.

Corresponding API Resources

"GET https://www.pantherdb.org/services/oai/pantherdb/familyortholog"
"GET https://www.pantherdb.org/services/oai/pantherdb/familymsa"
"GET https://www.pantherdb.org/services/oai/pantherdb/treeinfo"

References

- PANTHER Services Details
- Citations note on PANTHER website
rba_panther_homolog

See Also

Other "PANTHER": rba_panther_enrich(), rba_panther_homolog(), rba_panther_info(), rba_panther_mapping(), rba_panther_ortholog(), rba_panther_tree_grafter()

Examples

rba_panther_family("PTHR10000", what = "ortholog")

rba_panther_homolog (genes, organism, type = "P", target_organisms = NULL, ...)

Description

Using this function you can search and retrieve homolog of given gene(s).

Usage

rba_panther_homolog(genes, organism, type = "P", target_organisms = NULL, ...)

Arguments

genes Character vector of genes identifiers with maximum length of 10 or only one if seq_pos is supplied. Can be any of: Ensemble gene ID, Ensemble protein ID, Ensemble transcript ID, Entrez gene ID, gene symbol, NCBI GI, HGNC ID, International protein index ID, NCBI UniGene ID, UniProt accession and/or UniProt ID.

organism (numeric) NCBI taxon ID of the organism of your supplied genes. run rba_panther_info with argument 'what = "organisms"' to get a list of PANTHER's supported organisms.

type Homolog types to return. either "P" (default) for paralogs, "X" for horizontal gene transfer and "LDX" for diverged horizontal gene transfer.

target_organisms (numeric) NCBI taxon ID(s) to filter the results. run rba_panther_info with argument 'what = "organisms"' to get a list of PANTHER's supported organisms. For Paralog, target organism and organism should be the same; Otherwise, the target organism should be different from the input organism.

... rbioapi option(s). See rba_options's arguments manual for more information on available options.

Value

A dataframe with homologs information.
Corresponding API Resources

"GET https://www.pantherdb.org/services/oai/pantherdb/ortholog/homologOther"

References

- PANTHER Services Details
- Citations note on PANTHER website

See Also

Other "PANTHER": rba_panther_enrich(), rba_panther_family(), rba_panther_info(), rba_panther_mapping(), rba_panther_ortholog(), rba_panther_tree_grafter()

Examples

rba_panther_homolog("OR4F5", organism = 9606, type = "P")

rba_panther_info

Get PANTHER database Information

Description

Using this function you can retrieve a list of available organisms, annotation datasets, families, and pathways which are supported in PANTHER.

Usage

rba_panther_info(what, organism_chr_loc = FALSE, families_page = 1, ...)

Arguments

what what information to retrieve? should be one of:
- "organisms": Retrieve supported organisms in PANTHER.
- "datasets": Retrieve available annotation datasets.
- "families": Retrieve available family IDs.
- "species_tree": Retrieve the PANTHER's species tree.
- "pathways": Retrieve available pathway IDs.
organism_chr_loc

(Logical) (only when `what = "organisms"`) If TRUE, only organisms with chromosome location will be returned. If FALSE (default) every organisms will be returned.

families_page

(Numeric) (only when `what = "families"`) Family information is very long, so results are paginated. Use this argument to define the page to retrieve.

... rbioapi option(s). See `rba_options`'s arguments manual for more information on available options.

Value

For families and species tree, a list and otherwise a data frame with pertinent information.

Corresponding API Resources

"GET https://www.pantherdb.org/services/oai/pantherdb/supportedgenomes"
"GET https://www.pantherdb.org/services/oai/pantherdb/supportedannotdatasets"
"GET https://www.pantherdb.org/services/oai/pantherdb/supportedpantherfamilies"
"GET https://www.pantherdb.org/services/oai/pantherdb/supportedpantherpathways"
"GET https://www.pantherdb.org/services/oai/pantherdb/speciestree"

References

- PANTHER Services Details
- Citations note on PANTHER website

See Also

Other "PANTHER": `rba_panther_enrich()`, `rba_panther_family()`, `rba_panther_homolog()`, `rba_panther_mapping()`, `rba_panther_ortholog()`, `rba_panther_tree_grafter()`

Examples

```r
rba_panther_info(what = "organisms")

rba_panther_info(what = "families", families_page = 4)
```
Map A Gene-set to PANTHER Database

Description

Using this function, you can search your genes in PANTHER database and retrieve attributes and annotations associated to your genes.

Usage

rba_panther_mapping(genes, organism, ...)

Arguments

- **genes**: Character vector of genes identifiers with maximum length of 1000. Can be any of: Ensemble gene ID, Ensemble protein ID, Ensemble transcript ID, Entrez gene ID, gene symbol, NCBI GI, HGNC ID, International protein index ID, NCBI UniGene ID, UniProt accession and/or UniProt ID.
- **organism**: (numeric) NCBI taxon ID. run *rba_panther_info* with argument 'what = "organisms"' to get a list of PANTHER's supported organisms.
- **...**: rbioapi option(s). See *rba_options*’s arguments manual for more information on available options.

Value

A list containing your unmapped inputs and mapped genes with pertinent information.

Corresponding API Resources

"GET https://www.pantherdb.org/services/oai/pantherdb/geneinfo"

References

- PANTHER Services Details
- Citations note on PANTHER website

See Also

Other "PANTHER": *rba_panther_enrich()*, *rba_panther_family()*, *rba_panther_homolog()*, *rba_panther_info()*, *rba_panther_ortholog()*, *rba_panther_tree_grafter()"
Examples

```r
rba_panther_mapping(genes = c("Cd40", 7124, "ENSG00000203747", "P33681"),
                     organism = 9606)
```

Description

Using this function you can search and retrieve orthologs of given gene(s), and optionally return the corresponding position in the target organisms' protein sequences.

Usage

```r
rba_panther_ortholog(
  genes,
  organism,
  type = "all",
  target_organisms = NULL,
  seq_pos = NULL,
  include_msa = NULL,
  ...
)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>genes</td>
<td>Character vector of genes identifiers with maximum length of 10 or only one if seq_pos is supplied. Can be any of: Ensemble gene ID, Ensemble protein ID, Ensemble transcript ID, Entrez gene ID, gene symbol, NCBI GI, HGNC ID, International protein index ID, NCBI UniGene ID, UniProt accession and/or UniProt ID.</td>
</tr>
<tr>
<td>organism</td>
<td>(numeric) NCBI taxon ID of the organism of your supplied genes. run <code>rba_panther_info</code> with argument <code>what = &quot;organisms&quot;</code> to get a list of PANTHER’s supported organisms.</td>
</tr>
<tr>
<td>type</td>
<td>Ortholog types to return. either &quot;all&quot; (default) or &quot;LDO&quot; to only return least diverged orthologs.</td>
</tr>
<tr>
<td>target_organisms</td>
<td>(numeric) NCBI taxon ID(s) to filter the results. run <code>rba_panther_info</code> with argument <code>what = &quot;organisms&quot;</code> to get a list of PANTHER’s supported organisms.</td>
</tr>
<tr>
<td>seq_pos</td>
<td>(Numeric) A position in the protein’s sequence of the supplied gene. should be in the range of the protein’s length.</td>
</tr>
</tbody>
</table>
include_msa  (Logical) Only if a sequence position is supplied, should MSA (Multiple Sequence Alignment) information be included in the results?

Value

A data frame with Orthologs information.

Corresponding API Resources

"POST https://www.pantherdb.org/services/oai/pantherdb/ortholog/matchortho"
"POST https://www.pantherdb.org/services/oai/pantherdb/ortholog/homologpos"

References

• PANTHER Services Details
• Citations note on PANTHER website

See Also

Other "PANTHER": rba_panther_enrich(), rba_panther_family(), rba_panther_homolog(), rba_panther_info(), rba_panther_mapping(), rba_panther_tree_grafter()

Examples

```r
rba_panther_ortholog("CD40", organism = 9606, type = "LDO")
```

---

**rba_panther_tree_grafter**

*PANTHER Tree Grafter* Use this function to retrieve a PANTHER family’s tree topology information with a node corresponding to your sequence grafted in the best location in that tree.

Description

For more information, see: TreeGrafter: phylogenetic tree-based annotation of proteins with Gene Ontology terms and other annotations.
Usage

```r
rba_panther_tree_grafter(protein_seq, target_organisms = NULL, ...)
```

Arguments

- **protein_seq**  
  A character string with the protein's sequence. Maximum allowed sequence length is 50kb.

- **target_organisms**  
  (numeric) NCBI taxon ID(s) to filter the results. run `rba_panther_info` with argument 'what = "organisms"' to get a list of PANTHER's supported organisms.

- ...  
  rbioapi option(s). See `rba_options`'s arguments manual for more information on available options.

Value

A list containing PANTHER tree topology information.

Corresponding API Resources

"GET https://www.pantherdb.org/services/oai/pantherdb/graftsequence"

References


- PANTHER Services Details
- Citations note on PANTHER website

See Also

Other "PANTHER": `rba_panther_enrich()`, `rba_panther_family()`, `rba_panther_homolog()`, `rba_panther_info()`, `rba_panther_mapping()`, `rba_panther_ortholog()`

Examples

```r
rba_panther_tree_grafter("MKVLWAALLVTFLAGCQAKVEQAVETE")
```
**rba_reactome_analysis**  
*Reactome Over-Representation or Expression Analysis*

**Description**

Using this function, you can perform Reactome Analysis in a convenient way. The Analysis Type will be chosen depending on your supplied input:

1. If you supply a vector or a single-columned table, "Over-Representation" analysis will be performed.
2. If you supply a multi-column table, with the first column being molecules identifiers and the rest being numeral expression values, "Expression" analysis will be performed.

See the details section for the accepted input types and format.

**Usage**

```r
rba_reactome_analysis(
  input,
  input_format = NULL,
  projection = TRUE,
  interactors = FALSE,
  species = NULL,
  sort_by = "ENTITIES_PVALUE",
  order = "ASC",
  resource = "TOTAL",
  p_value = 1,
  include_disease = TRUE,
  min = NULL,
  max = NULL,
  ...
)
```

**Arguments**

- **input**  
  A vector, data frame, matrix or a local file path or URL that points to your data. See "Details section" for more information of how to organize and supply your input.

- **input_format**  
  (Optional) This function will automatically identify your supplied input’s format. But in case of unexpected issues or if you want to be explicit, set this argument to one of:
  - "table": If you supplied a data frame or matrix as input.
  - "vector": If you supplied a simple vector (numeric or character) as input.
  - "file": If you supplied a local file path pointing to a correctly-formatted text file.
  - "url": If you supplied a URL pointing to a correctly-formatted text file.
projection Logical (default = TRUE) Should non-human identifiers be projected to their human equivalents? (using Reactome orthology data)
interactors Logical (default = FALSE) Should IntAct interaction data be used to increase the analysis background?
species Numeric or Character: NCBI Taxonomy identifier (Human is 9606), species name (e.g. "Homo sapiens") or Reactome DbId (e.g Homo sapiens is 48887). See rba_reactome_species or Reactome Data Schema: Entries: Species. Note that you cannot supply the species parameter when projection parameter is TRUE.
sort_by Sort the result based on what column? available choices are: "NAME", "TOTAL_ENTITIES", "TOTAL_INTERACTORS", "TOTAL_REACTIONS", "FOUND_ENTITIES", "FOUND_INTERACTORS", "FOUND_REACTIONS", "ENTITIES_RATIO", "ENTITIES_PVALUE", "ENTITIES_FDR" or "REACTIONS_RATIO"
order Sort Order. Can be either "ASC" (default) or "DESC".
resource Filter results based on the resource. Default is "TOTAL", available choices are: "TOTAL", "UNIPROT", "ENSEMBL", "CHEBI", "IUPHAR", "MIRBASE", "NCBI_PROTEIN", "EMBL", "COMPOUND", "ENTITIES_FDR" or "PUBCHEM_COMPOUND".
p_value Set a P value threshold. Only results with P value equal to or less than your supplied threshold will be returned. (default = 1, Meaning no P value filtering)
include_disease Logical (default = TRUE) Should the disease pathways be included in the results?
min (numeric) Minimum number of entities that a pathways should have to be included in the results.
max (numeric) Maximum number of entities that a pathways should have to be included in the results.
... rbioapi option(s). See rba_options's arguments manual for more information on available options.

**Details**

You can supply your table or vector input in numerous formats:

1. A R object which can be data frame, matrix or a simple vector.
2. A path to a local text file in your device that contains the molecules data. (The file should be formatted correctly, see below.)
3. A URL pointing to a text file on the web that contains the molecules data. (The file should be formatted correctly, see below.)

If you supply a text file (as a local file path or URL), it should be in TSV (Tab-Separated Values) format; Column names should start with "#" character. Note that if you are providing the file for "Over-Representation" analysis (i.e. Single columned-data) this header line is optional and will be used as your 'Sample Name', otherwise it is required.

Also, form the "summary" element in the function’s output, you can see how Reactome Interpreted your input and subsequently the type of analysis that has been performed.

There is no strict criteria about the type of your molecules Identifiers, Reactome will Map the IDs
to it’s internal database entities. Nevertheless, You can check if all your identifiers has been found in "identifiersNotFound" element in the function’s output.

After Any Analysis, Reactome will associate a token to your analysis. It can be later used to in function that requires the token (e.g to retrieve the analysis results, download pdf).

Note that Reactome will store your token for only 7 days. You can download your full results with rba_reactome_analysis_download, and re-import it anytime to reactome (using rba_reactome_analysis_import) to generate a new token.

**Value**

List containing the results and information of your analysis. Note that you can use the token returned in the "summary" sub-list of the results (i.e. results$summary$token) to retrieve your results later or in other Reactome analysis functions.

**Corresponding API Resources**

"POST https://reactome.org/AnalysisService/identifiers/form"
"POST https://reactome.org/AnalysisService/identifiers/url"
"POST https://reactome.org/AnalysisService/identifiers/form/projection"
"POST https://reactome.org/AnalysisService/identifiers/url/projection"

**References**

- Reactome Analysis Services API Documentation
- Citations note on Reactome website

**See Also**

Other "Reactome Analysis Service": rba_reactome_analysis_download(), rba_reactome_analysis_import(), rba_reactome_analysis_mapping(), rba_reactome_analysis_pdf(), rba_reactome_analysis_species(), rba_reactome_analysis_token()

**Examples**

```r
## Not run:
rba_reactome_analysis(input = c("p53", "BRCA1", "cdk2", "Q99835", "CDC42"))
```

## End(Not run)
rba_reactome_analysis_download

Download Different Reactome Analysis Results

Description

Based on the "request" argument, you can download different analysis results data associated with a given token.

Usage

rba_reactome_analysis_download(
  token,
  request,
  save_to = NULL,
  resource = "TOTAL",
  ...
)

Arguments

token | A token associated to your previous Reactome analysis.
request | What to download? Should be one of:
  • "found_ids": Download a CSV file containing the found user-supplied identifiers in the analysis associated with your supplied token and resource.
  • "not_found_ids": Download a CSV file containing the user-supplied Identifiers which has not been found in the analysis associated with your supplied token.
  • "pathways": Download a CSV file containing Pathway analysis results of the analysis associated with your supplied token and resource.
  • "results": Download a JSON file containing the complete analysis results associated with your supplied token.
  • "results_gz": Same as "results", but the output will be compress (gzipped).
save_to | NULL or Character:
  • NULL: Save the file to an automatically-generated path.
  • Character string: A valid file path to save the file to.
resource (Only when request is "found_ids" or "pathways") Filter results based on the resource. Default is "TOTAL", available choices are: "TOTAL", "UNIPROT", "ENSEMBL", "CHEBI", "IUPHAR", "MIRBASE", "NCBI_PROTEIN", "EMBL", "COMPOUND", "ENTITIES_FDR" or "PUBCHEM_COMPOUND".

Details

Token is associated to each Reactome analysis results and kept by Reactome for at least 7 days. You can locate it in rba_reactome_analysis's output, under a sub-list named "summary" (i.e. results$summary$token).

Use rba_reactome_analysis_pdf to save a full report in PDF format.

Value

NULL, a CSV, JSON or Gzipped JSON file will be saved to disk based on your input.

Corresponding API Resources

GET https://reactome.org/AnalysisService/download/{token}/entities/found/{resource}/{filename}.csv"
GET https://reactome.org/AnalysisService/download/{token}/entities/notfound/{filename}.csv"
GET https://reactome.org/AnalysisService/download/{token}/pathways/{resource}/{filename}.csv"
GET https://reactome.org/AnalysisService/download/{token}/result.json"
GET https://reactome.org/AnalysisService/download/{token}/result.json.gz"

References


See Also

rba_reactome_analysis_pdf rba_reactome_analysis

Other "Reactome Analysis Service": rba_reactome_analysis(), rba_reactome_analysis_import(), rba_reactome_analysis_mapping(), rba_reactome_analysis_pdf(), rba_reactome_analysis_species(), rba_reactome_analysis_token()
rba_reactome_analysis_import

Import Saved Analysis JSON to Reactome

Description

If you have a JSON file of analysis results (only obtained via `rba_reactome_analysis_download` with the result argument set to "results", or "results_gz"), you can import the results back to Reactome and retrieve a token.

This is useful when you want to use other Reactome services which require a token but you do not have a token or your token has been expired (i.e. more than 7 days passed from your analysis).

Usage

`rba_reactome_analysis_import(input, input_format = NULL, ...)`

Arguments

- **input**: A local file path or URL that points to your -optionally gzipped- JSON file.
- **input_format**: (Optional) This function will automatically identify your supplied input's format. But in case of unexpected issues or if you want to be explicit, set this argument to one of:
  - "file": If you supplied a local file path pointing to the JSON file.
  - "url": If you supplied a URL pointing to the JSON file.
- ... rbioapi option(s). See `rba_options`'s arguments manual for more information on available options.

Value

A list containing the new token and other information of your imported results.

Corresponding API Resources

"GET https://reactome.org/AnalysisService/import/"
"GET https://reactome.org/AnalysisService/import/form"
"GET https://reactome.org/AnalysisService/import/url"
rba_reactome_analysis_mapping

References

- Reactome Analysis Services API Documentation
- Citations note on Reactome website

See Also

Other "Reactome Analysis Service": rba_reactome_analysis(), rba_reactome_analysis_download(), rba_reactome_analysis_mapping(), rba_reactome_analysis_pdf(), rba_reactome_analysis_species(), rba_reactome_analysis_token()

Examples

```r
## Not run:
rba_reactome_analysis_import("c:/rbioapi/res.json")
## End(Not run)
## Not run:
## Not run:
rba_reactome_analysis_import("https://qaz.com/res.json.gz")
## End(Not run)
```

rba_reactome_analysis_mapping

Maps Molecule Identifiers

Description

Use this function to map molecule identifiers of different species to Reactome Identifiers.

Usage

```r
rba_reactome_analysis_mapping(
  input,
  input_format = NULL,
  projection = TRUE,
)```
interactors = FALSE,
...
)

Arguments

input A vector, local file path or URL that points to your identifiers list.

input_format (Optional) This function will automatically identify your supplied input’s format. But in case of unexpected issues or if you want to be explicit, set this argument to one of:

- "vector": If you supplied a simple vector (numeric or character) as input.
- "file": If you supplied a local file path pointing to a correctly-formatted text file.
- "url": If you supplied a URL pointing to a correctly-formatted text file.

projection Logical (default = TRUE) Should non-human identifiers be projected to their human equivalents? (using Reactome orthology data)

interactors Logical (default = FALSE) Should IntAct interaction data be included?
...

rbioapi option(s). See rba_options’s arguments manual for more information on available options.

Value

List containing your identifiers and the IDS and resources they are mapped to.

Corresponding API Resources

"GET https://reactome.org/AnalysisService/mapping"
"GET https://reactome.org/AnalysisService/mapping/form"
"GET https://reactome.org/AnalysisService/mapping/form/projection"
"GET https://reactome.org/AnalysisService/mapping"
"GET https://reactome.org/AnalysisService/mapping/url"
"GET https://reactome.org/AnalysisService/mapping/url/projection"

References

- Reactome Analysis Services API Documentation
- Citations note on Reactome website
rba_reactome_analysis_pdf

See Also

Other "Reactome Analysis Service": rba_reactome_analysis(), rba_reactome_analysis_download(), rba_reactome_analysis_import(), rba_reactome_analysis_pdf(), rba_reactome_analysis_species(), rba_reactome_analysis_token()

Examples

## Not run:
rba_reactome_analysis_mapping(c("Q8SQ34", "cd40"))

## End(Not run)

---

rba_reactome_analysis_pdf

*Generate PDF file with Reactome Analysis Results*

Description

Use this function to save a detailed report of your previous analysis (That you have done with rba_reactome_analysis). You need to supply a 'token' associated to your previous analysis.

Usage

rba_reactome_analysis_pdf(
  token,
  species,
  save_to = NULL,
  number = 25,
  resource = "TOTAL",
  diagram_profile = "Modern",
  analysis_profile = "Standard",
  fireworks_profile = "Barium Lithium",
  ...
)

Arguments

token A token associated to your previous Reactome analysis.

species Numeric or Character: NCBI Taxonomy identifier (Human Taxonomy ID is 9606.) or species name (e.g. "Homo sapiens"). See rba_reactome_species or Reactome Data Schema: Entries: Species.

save_to NULL or Character:

  • NULL: Save the file to an automatically-generated path.
  • Character string: A valid file path to save the file to.
number
   Numeric: Maximum number of the reported pathways. Cannot not be greater than 50.

resource
   Filter results based on the resource. Default is "TOTAL", available choices are:
   "TOTAL", "UNIPROT", "ENSEMBL", "CHEBI", "IUPHAR", "MIRBASE", 
   "NCBI_PROTEIN", "EMBL", "COMPOUND", "ENTITIES_FDR" or "PUBCHEM_COMPOUND".

diagram_profile
   Color profile of diagrams, should be either "Modern" (default) or "Standard".

analysis_profile
   Color profile of analysis, should be one of: "Standard" (default), "Strosobar" or 
   "Copper Plus".

fireworks_profile
   Color profile of overview diagram, should be one of: "Copper", "Copper Plus", 
   "Barium Lithium" or "calcium salts".

...  rbioapi option(s). See rba_options’s arguments manual for more information 
on available options.

Details

Token is associated to each Reactome analysis results and kept by Reactome for at least 7 days. 
You can locate it in rba_reactome_analysis's output, under a sub-list named "summary" (i.e. res-
ults$summary$token).
Note that Reactome will store your token for only 7 days. You can download your full results with 
rba_reactome_analysis_download, and re-import it anytime to reactome (using rba_reactome_analysis_import) 
to generate a new token. Use rba_reactome_analysis_download to save your results in other for-
mats.

Value

NULL, a PDF file will be saved to disk.

Corresponding API Resources

"GET https://reactome.org/AnalysisService/report/{token}/{species}/ {filename}.pdf"

References

• Marc Gillespie, Bijay Jassal, Ralf Stephan, Marija Milacic, Karen Rothfels, Andrea Senff-
Ribeiro, Johannes Griss, Cristoffer Sevilla, Lisa Matthews, Chuqiao Gong, Chuan Deng, 
Thawfeek Varusai, Eliot Ragueneau, Yusra Haider, Bruce May, Veronica Shamovsky, Joel 
Weiser, Timothy Brunson, Nasim Sanati, Liam Beckman, Xiang Shao, Antonio Fabregat, 
Konstantinos Sidiropoulos, Julieth Murillo, Guilherme Viteri, Justin Cook, Solomon Shorser, 
Gary Bader, Emek Demir, Chris Sander, Robin Haw, Guanning Wu, Lincoln Stein, Henning 
Hermjakob, Peter D’Eustachio, The reactome pathway knowledgebase 2022, Nucleic Acids 
Research, 2021;, kab1028, https://doi.org/10.1093/nar/gkab1028

• Griss J, Viteri G, Sidiropoulos K, Nguyen V, Fabregat A, Hermjakob H. ReactomeGSA - 
rba_reactome_analysis_species

- Reactome Analysis Services API Documentation
- Citations note on Reactome website

See Also

rba_reactome_analysis_download rba_reactome_analysis

Other "Reactome Analysis Service": rba_reactome_analysis(), rba_reactome_analysis_download(), rba_reactome_analysis_import(), rba_reactome_analysis_mapping(), rba_reactome_analysis_species(), rba_reactome_analysis_token()

Examples

```r
## Not run:
rba_reactome_analysis_pdf(token = "MjAyMDEwMTYwMTI3MjY1MjM%3D",
  species = 9606, save_to = "my_analysis.pdf")

## End(Not run)
```

---

### rba_reactome_analysis_species

*Compare Human Pathways with with Other Species*

#### Description

Use This function to Compare human's manually-curated pathways and computationally inferred pathways (orthologous) in other species.

#### Usage

```r
rba_reactome_analysis_species(
  species_dbid,
  sort_by = "ENTITIES_PVALUE",
  order = "ASC",
  resource = "TOTAL",
  p_value = 1,
  min = NULL,
  max = NULL,
  ...
)
```

#### Arguments

- **species_dbid** Numeric: Reactome DbId (e.g Mus musculus is 48892) of the species you want to compare with Homo sapiens. See rba_reactome_species or Reactome Data Schema: Entries: Species.
sort_by: Sort the result based on what column? available choices are: "NAME", "TOTAL_ENTITIES", "TOTAL_INTERACTORS", "TOTAL_REACTIONS", "FOUND_ENTITIES", "FOUND_INTERACTORS", "FOUND_REACTIONS", "ENTITIES_RATIO", "ENTITIES_PVALUE", "ENTITIES_FDR" or "REACTIONS_RATIO"

order: Sort Order. Can be either "ASC" (default) or "DESC".

resource: Filter results based on the resource. Default is "TOTAL", available choices are: "TOTAL", "UNIPROT", "ENSEMBL", "CHEBI", "IUPHAR", "MIRBASE", "NCBI_PROTEIN", "EMBL", "COMPOUND", "ENTITIES_FDR" or "PUBCHEM_COMPOUND".

p_value: Set a P value threshold. Only results with P value equal to or less than your supplied threshold will be returned. (default = 1, Meaning no P value filtering)

min: (numeric) Minimum number of entities that a pathways should have to be included in the results.

max: (numeric) Maximum number of entities that a pathways should have to be included in the results.

...: rbioapi option(s). See rba_options’s arguments manual for more information on available options.

Details

Reactome incorporate manually curated human reactions and PANTHER’s protein homology data to Computationally infer events in other eukaryotic species.

In version 73 (11 June 2020), using an orthology-based approach, Homo sapiens events was projected to 18,654 orthologous pathways (with 81,835 orthologous proteins) in 15 non-human species. See Reactome Computationally Inferred Events for more information.

Value

List with the results of the comparison.

Corresponding API Resources

"GET https://reactome.org/AnalysisService/species/homoSapiens/{species}" 

References


See Also

rba_reactome_orthology

Other "Reactome Analysis Service": rba_reactome_analysis(), rba_reactome_analysis_download(), rba_reactome_analysis_import(), rba_reactome_analysis_mapping(), rba_reactome_analysis_pdf(), rba_reactome_analysis_token()

Examples

rba_reactome_analysis_species(species_dbid = 48892)

docs(rba_reactome_analysis_token)

rba_reactome_analysis_token

Return the Results Associated with a Token

Description

Use a token generated After a Reactome analysis (via rba_reactome_analysis) to Retrieve the analysis results. The output format is identical to the returned object of rba_reactome_analysis.

Usage

rba_reactome_analysis_token(
  token,
  species,
  sort_by = "ENTITIES_PVALUE",
  order = "ASC",
  resource = "TOTAL",
  p_value = NULL,
  include_disease = TRUE,
  min = NULL,
  max = NULL,
  ...
)

Arguments

token A token associated to your previous Reactome analysis.
species Numeric or Character: NCBI Taxonomy identifier (Human is 9606), species name (e.g. "Homo sapiens") or Reactome DbId (e.g Homo sapiens is 48887). See rba_reactome_species or Reactome Data Schema: Entries: Species.
### Details

After Any Analysis, Reactome will associate a token to your analysis. It can be later used in function that requires the token (e.g., to retrieve the analysis results, download PDF). Note that Reactome will store your token for only 7 days. You can download your full results with `rba_reactome_analysis_download` and re-import it anytime to reactome (using `rba_reactome_analysis_import`) to generate a new token.

### Value

List containing the results and information of your analysis.

### Corresponding API Resources

"GET https://reactome.org/AnalysisService/token/{token}"
- Reactome Analysis Services API Documentation
- Citations note on Reactome website

See Also

rba_reactome_analysis

Other "Reactome Analysis Service": rba_reactome_analysis(), rba_reactome_analysis_download(), rba_reactome_analysis_import(), rba_reactome_analysis_mapping(), rba_reactome_analysis_pdf(), rba_reactome_analysis_species()

Examples

```r
## Not run:
rba_reactome_analysis_token(token = "MjAyMDEwMTYwMTI3MjY1MjM",
species = 9606)
## End(Not run)
```

---

rba_reactome_complex_list

*Get Complexes That Include a Molecule*

Description

This function will retrieve a list of complexes that include your supplied molecule as a component.

Usage

```r
rba_reactome_complex_list(id, resource, ...)
```

Arguments

- `id`: Molecule’s external Identifier
- `resource`: What is the resource of your supplied ID? see: Reactome External Identifiers
- `...`: rbioapi option(s). See `rba_options`'s arguments manual for more information on available options.

Value

Data frame where each row is a complex containing your supplied molecule and columns are pertinent information.
**Corresponding API Resources**

"GET https://reactome.org/ContentService/data/complexes/{resource}/ {identifier}"

**References**

- Reactome Content Services API Documentation
- Citations note on Reactome website

**See Also**

Other "Reactome Content Service - Physical Entity Queries": `rba_reactome_complex_subunits()`, `rba_reactome_entity_other_forms()`, `rba_reactome_participant_of()`

**Examples**

```r
rba_reactome_complex_list(id = "3845", resource = "NCBI Gene")
```

```r
rba_reactome_complex_list(id = "P00533", resource = "UniProt")
```

---

**Description**

This function will return a list of subunits which are participants of your supplied complex.

**Usage**

```r
rba_reactome_complex_subunits(complex_id, exclude_structures = FALSE, ...)
```
Arguments

complex_id  Reactome stable Identifier of the complex.
exclude_structures  (logical) Should the contained complexes and entity sets be excluded from the results? (default = FALSE)
...  rbioapi option(s). See rba_options’ s arguments manual for more information on available options.

Details

Subunits will be returned recursively; Which means that if a subunit was itself a complex, subunit of that complex will be also returned in the results.

Value

Data frame which each row is a subunit of your supplied complex and the columns are pertinent information of that subunit.

Corresponding API Resources

"GET https://reactome.org/ContentService/data/complex/{id}/subunits"

References

- Reactome Content Services API Documentation
- Citations note on Reactome website

See Also

Other "Reactome Content Service - Physical Entity Queries": rba_reactome_complex_list(), rba_reactome_entity_other_forms(), rba_reactome_participant_of()
Examples

```r
rba_reactome_complex_subunits(complex_id = "R-HSA-5674003",
   exclude_structures = FALSE)
```

```r
rba_reactome_complex_subunits(complex_id = "R-HSA-109783",
   exclude_structures = TRUE)
```

---

**rba_reactome_diseases**  
*Reactome Diseases*

**Description**

This function Retrieve a list of all diseases or disease DOIDs annotated in Reactome.

**Usage**

```r
rba_reactome_diseases(doid = FALSE, ...)
```

**Arguments**

- `doid` (logical) Return disease DOIDs instead of diseases? (default = FALSE)
- `...` rbioapi option(s). See `rba_options`'s arguments manual for more information on available options.

**Value**

Data frame containing all the disease annotation available at Reactome. If doid was set to TRUE, DOID info will be returned instead.

**Corresponding API Resources**

- "GET https://reactome.org/ContentService/data/diseases"
- "GET https://reactome.org/ContentService/data/diseases/doid"

**References**


• Reactome Content Services API Documentation

• Citations note on Reactome website

Examples

rba_reactome_diseases()

rba_reactome_diseases(doid = TRUE)

rba_reactome_entity_other_forms

Get Other forms of a Reactome Entity

Description

This function retrieve a list containing all other forms of your supplied Physical Entity ID.

Usage

rba_reactome_entity_other_forms(entity_id, ...)

Arguments

entity_id Reactome’s entity ID.
...

rbioapi option(s). See rba_options’s arguments manual for more information on available options.

Details

According to Reactome API documentation, "These other forms are Physical Entities that share the same Reference Entity identifier, e.g. PTEN H93R R-HSA-2318524 and PTEN C124R R-HSA-2317439 are two forms of PTEN."

Value

Data frame where each row is other forms of your supplied Entity ID and columns are pertinent information.

Corresponding API Resources

"GET https://reactome.org/ContentService/data/entity/[id]/otherForms"
References

- Reactome Content Services API Documentation
- Citations note on Reactome website

See Also

Other "Reactome Content Service - Physical Entity Queries": `rba_reactome_complex_list()`, `rba_reactome_complex_subunits()`, `rba_reactome_participant_of()`

Examples

```r
rba_reactome_entity_other_forms("R-HSA-199420")
```

---

**rba_reactome_event_ancestors**

*Get Reactome Events Ancestors*

Description

Along with Reactome's events hierarchy, this function will retrieve all the events beginning from your supplied event up to the "Top level Pathway". See "Details section" for more information.

Usage

```r
rba_reactome_event_ancestors(event_id, ...)
```

Arguments

- `event_id` : Reactome event's identifier.
- `...` : rbioapi option(s). See `rba_options`'s arguments manual for more information on available options.
**Details**

By Reactome's definition, Events are the building blocks of biological processes and could be of two main classes: "Pathway" or "Reaction-like events". The events are organized in a hierarchical structure; and each event could be child or parent to another event; The hierarchy will always begin with a "Top level pathway" event. Also note that a given event could be part of more that one hierarchies.

**Value**

List which every element is a Data frame listing your supplied event along with it’s ancestor events. Because any given event can be part of more than one pathway hierarchy, the list may contain multiple data frames.

**Corresponding API Resources**

"GET https://reactome.org/ContentService/data/event/{id}/ancestors"

**References**


- Reactome Content Services API Documentation

- Citations note on Reactome website

**See Also**

Other "Reactome Content Service - Queries Related to Events": rba_reactome_event_hierarchy()

**Examples**

rba_reactome_event_ancestors("R-HSA-5673001")
Description

This function will retrieve the full Events hierarchy of your supplied species. Directly under each species, each child element is a "top Level Pathway". You can traverse the events tree down by following the "children" element.

Usage

rba_reactome_event_hierarchy(species, ...)

Arguments

- `species` Numeric or Character: NCBI Taxonomy identifier (Human Taxonomy ID is 9606.) or species name (e.g. "Homo sapiens"). See rba_reactome_species or Reactome Data Schema: Entries: Species.

- `...` rbioapi option(s). See rba_options’s arguments manual for more information on available options.

Details

By Reactome’s definition, Events are the building blocks of biological processes and could be of two main classes: "Pathway" or "Reaction-like events". The events are organized in a hierarchical structure; and each event could be child or parent to another event; The hierarchy will always begin with a "Top level pathway" event. Also note that a given event could be part of more that one hierarchies.

Value

List which is a representation of the species’s events hierarchy described in the "Details section".

Corresponding API Resources

"GET https://reactome.org/ContentService/data/eventsHierarchy/{species}"
rba_reactome_exporter_diagram

- Reactome Content Services API Documentation
- Citations note on Reactome website

See Also

Other "Reactome Content Service - Queries Related to Events": rba_reactome_event_ancestors()

Examples

```r
## Not run:
#very large response!
rba_reactome_event_hierarchy("Homo sapiens")
## End(Not run)
## Not run:
#very large response!
rba_reactome_event_hierarchy(9606)
## End(Not run)
```

rba_reactome_exporter_diagram

*Get a Reactome Event Diagram*

Description

This function could be called in two scenarios:

1. With `create_document = FALSE`: To retrieve an image of that event’s Diagram.
2. With `create_document = TRUE`: To retrieve a PDF document with the event’s diagram image and additional information.

see "Details section" for more information

Usage

```r
rba_reactome_exporter_diagram(
  event_id,
  save_to = NULL,
  create_document = FALSE,
  resource = "TOTAL",
  diagram_profile = "Modern",
  analysis_profile = "Standard",
  token = NULL,
)```
exp_column = NULL,
document_level = 1,
output_format = "png",
image_quality = 5,
flag_element = NULL,
flg_interactors = TRUE,
sel = NULL,
title = TRUE,
margin = 15,
ehld = FALSE,
...

Arguments

event_id  Reactome event's identifier.

save_to  NULL or Character:
  • NULL: Save the file to an automatically-generated path.
  • Character string: A valid file path to save the file to.

create_document  logical: Create PDF document instead of image? (default = FALSE)

resource  The analysis resource for which the results will be overlaid on top of the given pathways overview.

diagram_profile  Color profile of diagrams, should be either "Modern" (default) or "Standard".

analysis_profile  Color profile of analysis, should be one of: "Standard" (default), "Strosobar" or "Copper Plus"

token  The analysis Token for which the results will be overlaid on top of the given pathways overview. see: rba_reactome_analysis.

exp_column  numeric: (only if token is supplied) Specify the expression column for the overlay.

document_level  numeric: (Only if "create_document" is TRUE) if 0 (default) the event’s children will not be included in the PDF document. Set this to 1 to include event’s children.

output_format  (Only if "create_document" is FALSE) Image format of the saved diagram. Can be one of: png (default), jpeg, svg or gif.

image_quality  Numeric: (Only if "create_document" is FALSE), a number ranging from 1 to 10. 1 is the lowest quality and 10 is the highest (default = 5).

flag_element  (Only if "create_document" is FALSE) gene name, protein ID, chemical ID or Reactome ID of a diagram’s element to be flagged.

flg_interactors  Logical: (Only if "create_document" is FALSE) Should the interactor be considered when flagging a diagram element? (default = TRUE)
sel  (Only if "create_document" is FALSE) CSV line for highlighting element(s) selection in the diagram.

title  Logical: (Only if "create_document" is FALSE) Should the pathway name be displayed below the image? (default = TRUE)

margin  Numeric: (Only if "create_document" is FALSE) A number ranging from 0 to 20 to set as the image’s margin. (default = 15)

ehld  logical: (Only if "create_document" is FALSE) Should "Enhanced High Level Diagram" be considered?

...  rbioapi option(s). See rba_options’s arguments manual for more information on available options.

Details

If the function is called with create_document = FALSE:
The result will be an image with the format supplied in "output_format" argument. If the supplied event ID refers to a pathway, the image’s content will be the that pathways diagram. If the supplied event ID refers to a sub-pathway or reaction event, the parent pathway’s diagram will be exported, with that reaction or sub-pathway’s events highlighted.
Note that to export an image of reaction-like event separately, you should use rba_reactome_exporter_reaction.

If the function is called with create_document = TRUE:
A PDF document will contain an image of the event’s diagram and the following information of that events: Summation, Literature references, Edit history type, location, compartments and diseases.
note that if you call the function with "document level = 1", information of your supplied event’s children will also be included.

Value

NULL, Based to the inputs, an image or PDF file will be saved to disk.

Corresponding API Resources

"GET https://reactome.org/ContentService/exporter/diagram/{identifier} .{ext}"
"GET https://reactome.org/ContentService/exporter/document/event/ {identifier}.pdf"

References


• Reactome Content Services API Documentation
rba_reactome_exporter_event

Exports A Reactome Event to SBGN or SBML

Description

This function will export a supplied Reactome Event (Pathway or Reaction) to a SBGN (Systems Biology Graphical Notation) or SBML (Systems Biology Markup Language)

Usage

rba_reactome_exporter_event(event_id, output_format, save_to = NULL, ...)

See Also

rba_reactome_exporter_reaction rba_reactome_analysis

Other "Reactome Content Service - Format Exporter": rba_reactome_exporter_event(), rba_reactome_exporter_overview()

Examples

## Not run:
 rba_reactome_exporter_diagram(event_id = "R-HSA-177929",
 create_document = FALSE)

## End(Not run)
## Not run:
## Not run:
 rba_reactome_exporter_diagram(event_id = "R-HSA-6787403",
 create_document = FALSE)

## End(Not run)
## Not run:
## Not run:
 rba_reactome_exporter_diagram(event_id = "R-HSA-177929",
 create_document = TRUE)

## End(Not run)
## Not run:
## Not run:
 rba_reactome_exporter_diagram(event_id = "R-HSA-177929",
 output_format = "svg",
 save_to = "reactome_event_diagram.svg")

## End(Not run)
Arguments

- **event_id**: Reactome event's database IDs (DbId) or Stable IDs (StId).
- **output_format**: Either "sbgn" or "sbml".
- **save_to**: NULL or Character:
  - NULL: Save the file to an automatically-generated path.
  - Character string: A valid file path to save the file to.

... rbioapi option(s). See `rba_options`'s arguments manual for more information on available options.

Value

NULL, According to the inputs, a SBGN or SBML file will be saved to disk.

Corresponding API Resources

- "GET https://reactome.org/ContentService/exporter/event/ {identifier}.sbgn"
- "GET https://reactome.org/ContentService/exporter/event/ {identifier}.sbml"

References

- Reactome Content Services API Documentation
- Citations note on Reactome website

See Also

Other "Reactome Content Service - Format Exporter": `rba_reactome_exporter_diagram()`, `rba_reactome_exporter_overview()`, `rba_reactome_exporter_reaction()`

Examples

```r
## Not run:
rba_reactome_exporter_event(event_id = "R-HSA-177929",
                           output_format = "sbgn",
                           save_to = "R-HSA-177929.sbgn")
## End(Not run)
```
rba_reactome_exporter_overview

Get a Reactome Pathway Overview

Description

This function will Save a Pathway Overview of the supplied specie as an image file.

Usage

rba_reactome_exporter_overview(
  species,
  output_format = "png",
  save_to = NULL,
  image_quality = 5,
  flag_element = NULL,
  flg_interactors = TRUE,
  sel = NULL,
  title = TRUE,
  margin = 15,
  diagram_profile = "Copper",
  token = NULL,
  resource = "TOTAL",
  exp_column = NULL,
  coverage = FALSE,
  ...
)

Arguments

species Numeric or Character: NCBI Taxonomy identifier (Human Taxonomy ID is 9606.) or species name (e.g. "Homo sapiens"). See rba_reactome_species or Reactome Data Schema: Entries: Species.

output_format Images format, Can be one of: png (default), jpeg, svg or gif.

save_to NULL or Character:
  • NULL: Save the file to an automatically-generated path.
  • Character string: A valid file path to save the file to.

image_quality Numeric: A number ranging from 1 to 10. 1 is the lowest quality and 10 is the highest (default = 5).

## Not run:
rba_reactome_exporter_event(event_id = "R-HSA-177929",
  output_format = "sbgn")
## End(Not run)
flag_element  Gene name, protein ID, chemical ID or Reactome ID of a diagram’s element to be flagged.
flg_interactors Logical: Should the interactor be considered when flagging a diagram element? (default = TRUE)
set  CSV line for highlighting element(s) selection in the diagram.
title Logical: Should the pathway name be displayed below the image? (default = TRUE)
margin Numeric: A number ranging from 0 to 20 to set as the image’s margin. (default = 15)
diagram_profile Color profile of diagrams, should be one of "Copper" (default), "Copper Plus", "Barium Lithium" or "calcium salts".
token The analysis Token for which the results will be overlaid on top of the given pathways overview. see: rba_reactome_analysis.
resource The analysis resource for which the results will be overlaid on top of the given pathways overview.
exp_column numeric: (only if token is supplied) Specify the expression column for the overlay.
coverage Logical: Should the analysis coverage values be overlaid? (default = FALSE)
... rBioapi option(s). See rba_options’s arguments manual for more information on available options.

Value
NULL, Based to the inputs, an image file will be saved to disk.

Corresponding API Resources
"GET https://reactome.org/ContentService/exporter/fireworks/{species}.{ext}"
See Also

rba_reactome_analysis

Other "Reactome Content Service - Format Exporter": rba_reactome_exporter_diagram(), rba_reactome_exporter_event(), rba_reactome_exporter_reaction()

Examples

## Not run:
```r
rba_reactome_exporter_overview(species = 9606,
                               output_format = "svg",
                               save_to = "human_pathways.svg")
```
## End(Not run)

## Not run:
```r
rba_reactome_exporter_overview(species = 9606,
                               token = 123456789)
```
## End(Not run)

---

rba_reactome_exporter_reaction

*Get a Reactome Reaction Event*

Description

This function will Save a Reactome event of class "ReactionLikeEvent" as an image file.

Usage

```r
rba_reactome_exporter_reaction(
  event_id,
  save_to = NULL,
  output_format = "png",
  resource = "TOTAL",
  diagram_profile = "Modern",
  analysis_profile = "Standard",
  token = NULL,
  exp_column = NULL,
  image_quality = 5,
  flag_element = NULL,
  flg_interactors = TRUE,
  sel = NULL,
  title = TRUE,
  margin = 15,
  ...
)
```
**Arguments**

**event_id**
- Reactome Reaction-like event’s identifier.

**save_to**
- NULL or Character:
  - NULL: Save the file to an automatically-generated path.
  - Character string: A valid file path to save the file to.

**output_format**
- Images format, Can be one of: png (default), jpeg, svg or gif.

**resource**
- The analysis resource for which the results will be overlaid on top of the given pathways overview.

**diagram_profile**
- Color profile of diagrams, should be one of "Copper" (default), "Copper Plus", "Barium Lithium" or "calcium salts".

**analysis_profile**
- Color profile of analysis, should be one of: "Standard" (default), "Strosobar" or "Copper Plus".

**token**
- The analysis Token for which the results will be overlaid on top of the given pathways overview. see: rba_reactome_analysis.

**exp_column**
- numeric: (only if token is supplied) Specify the expression column for the overlay.

**image_quality**
- Numeric: A number ranging from 1 to 10. 1 is the lowest quality and 10 is the highest (default = 5).

**flag_element**
- Gene name, protein ID, chemical ID or Reactome ID of a diagram’s element to be flagged.

**flg_interactors**
- Logical: Should the interactor be considered when flagging a diagram element? (default = TRUE)

**sel**
- CSV line for highlighting element(s) selection in the diagram.

**title**
- Logical: Should the pathway name be displayed below the image? (default = TRUE)

**margin**
- Numeric: A number ranging from 0 to 20 to set as the image’s margin. (default = 15)

**...**
- rbioapi option(s). See rba_options’s arguments manual for more information on available options.

**Details**

Note that this function will save Reaction-like event separately and out of it’s parent pathway context. To overlay a Reaction on it’s parent pathway, use rba_reactome_exporter_diagram.

**Value**

NULL, Based to the inputs, an image file will be saved to disk.

**Corresponding API Resources**

"GET https://reactome.org/ContentService/exporter/reaction/ {identifier}.{ext}"
References

- Reactome Content Services API Documentation
- Citations note on Reactome website

See Also

rba_reactome_exporter_diagram rba_reactome_analysis

Other "Reactome Content Service - Format Exporter": rba_reactome_exporter_diagram(), rba_reactome_exporter_event(), rba_reactome_exporter_overview()

Examples

```r
## Not run:
rba_reactome_exporter_diagram(event_id = "R-HSA-6787403",
create_document = FALSE)

## End(Not run)
## Not run:
rba_reactome_exporter_diagram(event_id = "R-HSA-6787403",
output_format = "svg",
save_to = "reactome_reacion_image.svg")

## End(Not run)
```

rba_reactome_interactors_psicquic

The interface From Reactome to PSICQUIC

Description

You can call this function in two scenarios: 1- To retrieve information of all available PSICQUIC resources, call the function without providing any argument; i.e rba_reactome_interactors_psicquic(). 2-To retrieve a list of interactors of specific protein(s), fill out the function’s arguments.
Usage

```r
rba_reactome_interactors_psicquic(
  proteins = NULL,
  resource = NULL,
  details = TRUE,
  ...
)
```

Arguments

- **proteins**: Proteins to retrieve PSICQUIC interactors.
- **resource**: The PSICQUIC resource for your supplied proteins. Call `rba_reactome_interactors_psicquic()` without argument to get the available options.
- **details**: Logical: If TRUE (default) a detailed list of interactors will be returned. If FALSE, only a summary of available interactors will be returned.
- **...**: `rbioapi` option(s). See `rba_options`’s arguments manual for more information on available options.

Value

Depending your input, a list containing the detailed or summary of PSICQUIC interactions or a data frame of all registered PSICQUIC resources.

Corresponding API Resources

- "POST https://reactome.org/ContentService/interactors/psicquic/molecules/ {resource}/details"
- "POST https://reactome.org/ContentService/interactors/psicquic/molecules/ {resource}/summary"
- "GET https://reactome.org/ContentService/interactors/psicquic/resources"

References

- Reactome Content Services API Documentation
- Citations note on Reactome website

See Also

Other "Reactome Content Service - Molecule Interactors": `rba_reactome_interactors_static()`
Examples

```r
rba_reactome_interactors_psicquic()

rba_reactome_interactors_psicquic(proteins = c("TP53", "MYC"),
    resource = "BioGrid",
    details = FALSE)

rba_reactome_interactors_psicquic(proteins = c("TP53", "MYC"),
    resource = "BioGrid",
    details = TRUE)
```

---

**rba_reactome_interactors_static**

*Get Static(IntAct) Interaction Information of a Protein*

**Description**

Reactome maintain a locally host a version of IntAct(Static) interactions database. Using this function, you can retrieve IntAct information of a protein(s) in two scenarios:

1. If `endpoint = "details"` or "summary": Retrieve a detailed/summary information of your supplied protein accession(s) from IntAct database.
2. If `endpoint = "pathway"`, Retrieve a list of Reactome pathways which include your supplied protein accession. Pathways with the class "TopLevelPathway" will be excluded.

**Usage**

```r
rba_reactome_interactors_static(
    proteins,
    endpoint = "details",
    only_diagrammed = FALSE,
    species = NULL,
    ...
)
```

**Arguments**

- `proteins`: Uniprot proteins accession(s). If `endpoint = "pathway"`, only a single protein accession can be supplied.
- `endpoint`: Can be one of:
  1. "details": To return a detailed information of your supplied protein(s) accession.
2. "summary": To return a summary of your supplied protein(s) accession
3. "pathway": To return a list of pathways containing the interacting molecules (excluding TopLevelPathway class).

**only_diagrammed**

Logical: (only when "endpoint = "pathway") If TRUE, pathways without diagram will be excluded. (default = FALSE)

**species**

Only when "endpoint = "pathway", The scientific name of the species to search for the pathways. See rba_reactome_species or Reactome Data Schema: Entries: Species.

... rbioapi option(s). See rba_options’s arguments manual for more information on available options.

**Value**

List which it’s content varies based on the supplied "endpoint" argument.

**Corresponding API Resources**

"POST https://reactome.org/ContentService/interactors/static/molecules/details"
"POST https://reactome.org/ContentService/interactors/static/molecules/summary"
"GET https://reactome.org/ContentService/interactors/static/molecules/pathways"

**References**

- Reactome Content Services API Documentation
- Citations note on Reactome website

**See Also**

Other "Reactome Content Service - Molecule Interactors": rba_reactome_interactors_psicquic()
rba_reactome_mapping

Map External ID to Reactome Pathways/Reactions

Description

By providing an external identifier from a given resource, you can retrieve a list of pathways/reactions that include your supplied ID.

Usage

rba_reactome_mapping(id, resource, map_to, species = "Homo sapiens", ...)

Arguments

id Molecule’s external Identifier
resource What is the resource of your supplied ID? see: Reactome External Identifiers
map_to Either "pathways" or "reactions".
species Numeric or Character: NCBI Taxonomy identifier (Human is 9606), species name (e.g. "Homo sapiens") or Reactome DbId (e.g Homo sapiens is 48887). See rba_reactome_species or Reactome Data Schema: Entries: Species.
... rbioapi option(s). See rba_options’s arguments manual for more information on available options.

Value

Data frame where each row is a pathway/reaction and columns are pertinent information.

Corresponding API Resources

"GET https://reactome.org/ContentService/data/mapping/{resource}/ {identifier}/pathways"
"GET https://reactome.org/ContentService/data/mapping/{resource}/ {identifier}/reactions"
References

- Citations note on Reactome website

Examples

rba_reactome_mapping(id = "PTEN", resource = "UniProt", map_to = "reactions", species = 9606)

rba_reactome_orthology

Get Orthologous (Computationally Inferred) Events

Description

Reactome incorporate manually curated human reactions and PANTHER’s protein homology data to Computationally infer events in other eukaryotic species.

Usage

rba_reactome_orthology(event_ids, species_dbid, ...)

Arguments

- event_ids: Human Reactome event ID(s) to retrieve their orthologous events.
- species_dbid: Reactome database ID (DbId) of the target species. (e.g. Mus musculus is 48892). See rba_reactome_species or Reactome Data Schema: Entries: Species.
- ...: rbioapi option(s). See rba_options’s arguments manual for more information on available options.
Details

In version 73 (11 June 2020), using an orthology-based approach, Homo sapiens events was projected to 18,654 orthologous pathways (with 81,835 orthologous proteins) in 15 non-human species. See Reactome Computationally Inferred Events for more information.

Value

List containing found Orthologous event(s) in your supplied species and their pertinent information.

Corresponding API Resources

"POST https://reactome.org/ContentService/data/orthologies/ids/species/{speciesId}"

References


See Also

rba_reactome_analysis_species

Examples

rba_reactome_orthology(event_ids = c("R-HSA-6799198", "R-HSA-72764"),
species_dbid = 49633)
Description

Participating molecules in a Reactome comprises set of 'Physical Entity' and 'Reference Entities' class objects. Use this function to retrieve all, only 'Physical Entity' or only 'Reference Entities' participants of given event.

Usage

```r
rba_reactome_participants(
  event_id,
  only_physical_entities = FALSE,
  only_reference_entities = FALSE,
  ...
)
```

Arguments

- **event_id**: Reactome event's database ID (DbId) or Stable ID (StId).
- **only_physical_entities**: Logical: If TRUE, only participating 'Physical Entities' will be returned.
- **only_reference_entities**: Logical: If TRUE, only participating 'Reference Entities' will be returned.
- **...**: rbioapi option(s). See `rba_options`'s arguments manual for more information on available options.

Details

A 'Physical Entity' Instance could include an individual molecule, a multi-molecular complex or a set of molecule forming a group based on some characteristics. A single molecule can have different 'Physical Entity' instances based on its associated attributes. For example, IgK Ig kappa chain, has two 'Physical Entity' instances; one, with ID "R-HSA-197041" refers to the secreted antibody protein to the extra-cellular region; And the second one is with ID "R-HSA-2038819" and refers to the plasma-membrane-integrated form of the antibody protein.

To make it possible to link multiple 'Physical Entity' instances of a molecule, Reactome uses a data class named "'Reference Entities'" which correspond to the invariant attribute of a molecule. for example, both of the above-mentioned 'Physical Entities' see a 'Reference Entities' named "UniProt:P01834 IGKC.

See Reactome Data Model for more information about the data model and Physical Entities.

Value

List with the participant of your supplied Event ID. A Data frame if only physical or 'Reference Entities' was requested.
Corresponding API Resources

"GET https://reactome.org/ContentService/data/participants/{id}"
"GET https://reactome.org/ContentService/data/participants/{id}/participatingPhysicalEntities"
"GET https://reactome.org/ContentService/data/participants/{id}/referenceEntities"

References

- Reactome Content Services API Documentation
- Citations note on Reactome website

See Also

rba_reactome_participant_of

Examples

rba_reactome_participants("R-HSA-5682012")

rba_reactome_participants("R-HSA-5682012", only_physical_entities = TRUE)

rba_reactome_participants("R-HSA-5682012", only_reference_entities = TRUE)

Description

This function will retrieve a list of complexes and sets that Your supplied entity ID participates in (e.g. as a complex component, reaction output).
Usage

```
    rba_reactome_participant_of(entity_id, ...)
```

Arguments

- **entity_id**: Reactome's entity ID.
- **...**: `rbiapi` option(s). See `rba_options`'s arguments manual for more information on available options.

Value

List of Reactome database Entities which Your supplied ID is a participant in them.

Corresponding API Resources

"GET https://reactome.org/ContentService/data/entity/{id}/componentOf"

References

- Reactome Content Services API Documentation
- Citations note on Reactome website

See Also

```
    rba_reactome_participants
```

Other "Reactome Content Service - Physical Entity Queries":
```
    rba_reactome_complex_list(), rba_reactome_complex_subunits(), rba_reactome_entity_other_forms()
```

Examples

```
    rba_reactome_participant_of(entity_id = "R-HSA-199420")
```
rba_reactome_pathways_events

Get Events Contained in an Upstream Events

Description

A Reactome Event could be comprised of other events (meaning, a pathway that include other pathways itself). Use this function to recursively return all the events which reside downstream of your supplied event ID (or an attribute of that events).

Usage

rba_reactome_pathways_events(event_id, attribute_name = NULL, ...)

Arguments

- event_id: Reactome event’s database ID (DbId) or Stable ID (StId).
- attribute_name: An attribute of the events to be returned instead of the whole events. see Reactome Data Schema: Event for available options.
- ...: rbioapi option(s). See rba_options’s arguments manual for more information on available options.

Details

By Reactome’s definition, Events are the building blocks of biological processes and could be of two main classes: "Pathway" or "Reaction-like events". The events are organized in a hierarchical structure; and each event could be child or parent to another event; The hierarchy will always begin with a "Top level pathway" event. Also note that a given event could be part of more that one hierarchies.

Value

Data frame where each row is a contained event and columns are event’s attributes. If an "attribute_name" argument was supplied, a character vector will be returned.

Corresponding API Resources

"GET https://reactome.org/ContentService/data/pathway/{id}/ containedEvents"
"GET https://reactome.org/ContentService/data/pathway/{id}/ containedEvents/{attributeName}"

References


- Reactome Content Services API Documentation
- Citations note on Reactome website

See Also

Other "Reactome Content Service - Pathway Related Queries": rba_reactome_pathways_low(), rba_reactome_pathways_top()

Examples

```r
rba_reactome_pathways_events(event_id = "R-HSA-5673001")

rba_reactome_pathways_events(event_id = "R-HSA-5673001",
    attribute_name = "displayName")
```

---

rba_reactome_pathways_low

*Get lower level pathways Containing a 'Physical Entity' or Event*

Description

Use this function to search the event hierarchy and retrieve a list of all lower level pathways (non TopLevelPathway class) that contain a given 'Physical Entity' or Event. See "Arguments section" on how to modify your search.

Usage

```r
rba_reactome_pathways_low(
    entity_id,
    with_diagram = FALSE,
    all_forms = FALSE,
    species = NULL,
    ...
)
```
Arguments

- **entity_id**: The entity that should exist in the pathways.
- **with_diagram**: Logical: only include pathways with diagram?
- **all_forms**: Logical: should other variants of your supplied entity_id be considered? (e.g. same molecule but in different compartment, secretory form etc.) see rba_reactome_participants’s “Details section” to learn more about how Reactome classifies molecules.
- **species**: (optional) Numeric or Character: confine your search to a specific species by providing its NCBI Taxonomy identifier (Human Taxonomy ID is 9606) or species name (e.g. "Homo sapiens"). See rba_reactome_species or Reactome Data Schema: Entries: Species.

Value

Data frame where each row is a pathway that contains your supplied entity and columns are pertinent information.

Corresponding API Resources

"GET https://reactome.org/ContentService/data/pathways/low/entity/{id}"
"GET https://reactome.org/ContentService/data/pathways/low/diagram/ entity/{id}"
"GET https://reactome.org/ContentService/data/pathways/low/diagram/ entity/{id}/allForms"

References


See Also

Other "Reactome Content Service - Pathway Related Queries": rba_reactome_pathways_events(), rba_reactome_pathways_top()
Examples

rba_reactome_pathways_low(entity_id = "R-HSA-199420")

rba_reactome_pathways_low(entity_id = "R-HSA-199420", with_diagram = TRUE)

rba_reactome_pathways_low(entity_id = "R-HSA-199420", with_diagram = TRUE, all_forms = TRUE)

rba_reactome_pathways_top

Get Top Level Pathways in a Species

Description

This function will return a list of all pathways with the class "TopLevelPathway" which are annotated in your supplied species.

Usage

rba_reactome_pathways_top(species, ...)

Arguments

species Numeric or Character: NCBI Taxonomy identifier (Human Taxonomy ID is 9606.) or species name (e.g. "Homo sapiens"). See rba_reactome_species or Reactome Data Schema: Entries: Species.

... rbioapi option(s). See rba_options’s arguments manual for more information on available options.

Details

Reactome’s Events hierarchy for any species will begin with pathways with class "TopLevelPathway" (e.g. "Immune System", "Metabolism of proteins"). Further down in the event’s hierarchy tree, each TopLevelPathway has has other events itself (e.g. "Adaptive immune system", "Innate immune system"). Based on the chosen pathway, the hierarchy tree would typically goes further down.

Value

Data frame where each row is a Top Level Pathway and columns are pertinent information.

Corresponding API Resources

"GET https://reactome.org/ContentService/data/pathways/top/{species}"
References

- Reactome Content Services API Documentation
- Citations note on Reactome website

See Also

Other "Reactome Content Service - Pathway Related Queries": rba_reactome_pathways_events(), rba_reactome_pathways_low()

Examples

```r
rba_reactome_pathways_top(species = 9606)

rba_reactome_pathways_top(species = "Saccharomyces cerevisiae")
```

---

rba_reactome_people_id

A person by his identifiers

**Description**

A person by his identifiers

**Usage**

```r
rba_reactome_people_id(
    person_id,
    authored_pathways = FALSE,
    publications = FALSE,
    attribute_name = NULL,
    ...
)
```
Arguments

person_id  Reactome database ID (DbId) or ORCHID ID
authored_pathways  Logical: Only return Pathway list authored by the person? (default = FALSE)
publications  Logical: Only return publications list authored by the person? (Defalt = FALSE)
attribute_name  (optional) A Reactome person attribute to return only. see Reactome Data Schema: person for available options.

...  rbioapi option(s). See rba_options's arguments manual for more information on available options.

Value

List containing the requested informations of your supplied person.

Corresponding API Resources

"GET https://reactome.org/ContentService"

References

- Reactome Content Services API Documentation
- Citations note on Reactome website

See Also

Other "Reactome Content Service - Person Queries": rba.reactome_people_name()

Examples

rba.reactome_people_id("391309")

rba.reactome_people_id(person_id = "391309", authored_pathways = TRUE)
Get Persons Information by Name

Description

Using this function you can query people by partially matching or exact name and retrieve a list of matching people in Reactome.

Usage

```r
rba_reactome_people_name(person_name, exact_match = FALSE, ...)
```

Arguments

- `person_name` first and last name of the person
- `exact_match` Logical: should the supplied name be considered as an exact match? (default = FALSE)
- `...` rbioapi option(s). See `rba_options`’s arguments manual for more information on available options.

Value

List where each element is a search hit contains the person’s information.

Corresponding API Resources

"GET https://reactome.org/ContentService/data/people/name/{name}"  
"GET https://reactome.org/ContentService/data/people/name/{name}/exact"

References

- Reactome Content Services API Documentation
- Citations note on Reactome website
See Also

Other "Reactome Content Service - Person Queries": rba_reactome_people_id()

Examples

rba_reactome_people_name("Jupe")

rba_reactome_people_name("Steve Jupe", exact_match = TRUE)

Description

Using this Comprehensive function, You can Retrieve any object from Reactome knowledge-base

Usage

rba_reactome_query(
  ids,
  enhanced = FALSE,
  map = FALSE,
  attribute_name = NULL,
  ...
)

Arguments

ids A single or Multiple database IDs (DbId), Stable IDs (StId) or a mixture of both.
enhanced Logical: (Default = FALSE) If 'TRUE' more information on the supplied entry will be returned. (You can set this argument to 'TRUE' Only when you supply a single ID).
map (Default = FALSE) Should the supplied IDs be mapped? This argument will only be considered when you supply multiple IDs. (e.g. when you supply previous version of stable identifiers.)
attribute_name (Optional) Only Return an Attribute of the supplied Database Object. (You can use this argument Only when you supply a single ID)
...

Value

List containing your query outputs.
Corresponding API Resources

"POST https://reactome.org/ContentService/data/query/ids"
"POST https://reactome.org/ContentService/data/query/ids/map"
"GET https://reactome.org/ContentService/data/query/{id}" 
"GET https://reactome.org/ContentService/data/query/enhanced/{id}" 
"GET https://reactome.org/ContentService/data/query/{id}/[attributeName]"

References

- Reactome Content Services API Documentation
- Citations note on Reactome website

Examples

```r
rba_reactome_query(ids = c("8953958", "11982506", "R-ALL-9649879"))
```

```r
rba_reactome_query(ids = "R-HSA-9656256", enhanced = TRUE)
```

```r
rba_reactome_query(ids = "8863054", attribute_name = "displayName")
```

### rba_reactome_species

**Get Reactome Species**

**Description**

Use this function to retrieve a table of Available species in Reactome.

**Usage**

```r
rba_reactome_species(only_main = FALSE, ...)
```
Arguments

only_main Logical: If set to TRUE, will only return species which have either manually-curated or computationally inferred pathways.

... rbioapi option(s). See rba_options’s arguments manual for more information on available options.

Value

Data frame where each row is a species and columns are pertinent information.

Corresponding API Resources

"GET https://reactome.org/ContentService/data/species/all"
"GET https://reactome.org/ContentService/data/species/main"

References


• Reactome Content Services API Documentation

Examples

rba.reactome_species()

rba.reactome_species(only_main = TRUE)
rba_reactome_version  The version number of current database

Description

Returns the current version of Reactome database.

Usage

rba_reactome_version(...)  

Arguments

...  rbioapi option(s). See rba_options’s arguments manual for more information on available options.

Value

Character string containing the version of Reactome database.

Corresponding API Resources

"GET https://reactome.org/ContentService/data/database/version"

References


• Reactome Content Services API Documentation

Examples

rba_reactome_version()
**rba_reactome_xref**  
*Map Cross References IDs to Reactome ReferenceEntity*

---

**Description**

Use this function To retrieve a list of Reactome ReferenceEntity associated to your supplied Cross Reference (i.e. External) ID.

**Usage**

```r
rba_reactome_xref(xref_id, ...)  
```

**Arguments**

- `xref_id`: molecule's cross-reference (external) identifier.
- `...`: `rbioapi` option(s). See `rba_options`'s arguments manual for more information on available options.

**Details**

Reactome cross-references external database's identifiers to its database Entries named ReferenceEntity, which resembles the invariant aspect of a molecule. Thus there is a one-to-many relationship between Reactome’s ReferenceEntity object and the molecule’s ID in external databases, which in Reactome’s terms is called Cross Reference.  
See `rba_reactome_participants`'s "Details section" to learn more about how Reactome classifies molecules.

**Value**

List containing the ReferenceEntity corresponding to your supplied cross-reference (external) ID.

**Corresponding API Resources**

"GET https://reactome.org/ContentService/references/mapping/{identifier}"  

**References**

rba_string_annotations

Description

STRING cross-reference the proteins with several databases (see "Details" section). By providing your input set of proteins (and optionally background or universe protein set), you can use this function to retrieve full set of terms (annotations) pertinent to your input proteins in each database, among with information for each term.

Usage

rba_string_annotations(
  ids,
  species = NULL,
  allow_pubmed = FALSE,
  split_df = TRUE,
  ...
)

Arguments

ids Your protein ID(s). It is strongly recommended to supply STRING IDs. See rba_string_map_ids for more information.
species Numeric: NCBI Taxonomy identifier; Human Taxonomy ID is 9606. (Recommended, but optional if your input is less than 100 IDs.)
allow_pubmed logical: (default = FALSE) PubMed usually assigns a large number of reference publications to each protein. In order to reduce the output size, PubMed’s results will be excluded from the results, unless stated otherwise by setting this argument to TRUE.
split_df (logical, default = TRUE), If TRUE, instead of one data frame, results from different categories will be split into multiple data frames based on their ‘category’.
... rbioapi option(s). See rba_options’s arguments manual for more information on available options.
**rba_string_annotations**

**Details**

STRING currently maps to and retrieve enrichment results based on Gene Ontology (GO), KEGG pathways, UniProt Keywords, PubMed publications, Pfam domains, InterPro domains, and SMART domains. Note that this function will return a full list of the terms containing your supplied proteins. To perform enrichment and only retrieve a enriched subset of the terms, use `rba_string_enrichment`.

**Value**

A data frame which every row is an assigned terms and the columns are the terms category, description, number of genes, and other pertinent information.

**Corresponding API Resources**

"POST https://string-db.org/api/{output_format}/functional_annotation?identifiers={your_identifiers}&{optional_parameters}"

**References**

- STRING API Documentation
- Citations note on STRING website

**See Also**

`rba_string_map_ids`, `rba_string_enrichment`

Other "STRING": `rba_string_enrichment()`, `rba_string_enrichment_ppi()`, `rba_string_homology_inter()`, `rba_string_homology_intra()`, `rba_string_interaction_partners()`, `rba_string_interactions_network()`, `rba_string_map_ids()`, `rba_string_network_image()`, `rba_string_version()`

**Examples**

```r
rba_string_annotations(ids = "TP53", species = 9606)
```
rba_string_enrichment  Getting Functional Enrichment

Description
STRING cross-reference the proteins with several databases (see "Details" section). By providing your input set of proteins (and optionally background or universe protein set), you can use this function to perform enrichment test and retrieve a list of enriched terms in each database, among with pertinent information for each term.

Usage
rba_string_enrichment(
  ids,
  species = NULL,
  background = NULL,
  split_df = TRUE,
  ...
)

Arguments
ids  Your protein ID(s). It is strongly recommended to supply STRING IDs. See rba_string_map_ids for more information.

species Numeric: NCBI Taxonomy identifier; Human Taxonomy ID is 9606. (Recommended, but optional if your input is less than 100 IDs.)

background character vector: A set of STRING protein IDs to be used as the statistical background (or universe) when computing P-value for the terms. Only STRING IDs are acceptable. (See rba_string_map_ids to map your IDs.)

split_df (logical, default = TRUE). If TRUE, instead of one data frame, results from different categories will be split into multiple data frames based on their 'category'.

...  rbioapi option(s). See rba_options’s arguments manual for more information on available options.

Details
STRING currently maps to and retrieve enrichment results based on Gene Ontology (GO), KEGG pathways, UniProt Keywords, PubMed publications, Pfam domains, InterPro domains, and SMART domains.
Note that this function will only return the enriched terms pertinent to your proteins that have a p-value lesser than 0.1. To retrieve a full list of the terms -unfiltered by enrichment p-values-, use rba_string_annotations.
**Value**

A list of data frames which every row is an enriched terms with p-value smaller than 0.1 and the columns are the terms category, description, number of genes, p-value, fdr and other pertinent information.

**Corresponding API Resources**

"POST https://string-db.org/api/{output_format}/enrichment?identifiers= {your_identifiers}&{optional_parameters}"

**References**

- STRING API Documentation
- Citations note on STRING website

**See Also**

rba_string_map_ids, rba_string_annotations

Other "STRING": rba_string_annotations(), rba_string_enrichment_ppi(), rba_string_homology_inter(), rba_string_homology_intra(), rba_string_interaction_partners(), rba_string_interactions_network(), rba_string_map_ids(), rba_string_network_image(), rba_string_version()

**Examples**

```r
rba_string_enrichment(ids = c("TP53", "TNF", "EGFR"), species = 9606)
```

---

**rba_string_enrichment_ppi**

*Get Protein-Protein Interaction Enrichment*

**Description**

Even when there is no annotation for your input proteins, STRING can Compare your Given proteins interactions pattern with the background proteome-wide interaction distribution to determine if your given set of proteins are functionally related.
rba_string_enrichment_ppi

Usage

rba_string_enrichment_ppi(
  ids,
  species = NULL,
  required_score = NULL,
  background = NULL,
  ...
)

Arguments

ids  Your protein ID(s). It is strongly recommended to supply STRING IDs. See rba_string_map_ids for more information.

species  Numeric: NCBI Taxonomy identifier; Human Taxonomy ID is 9606. (Recommended, but optional if your input is less than 100 IDs.)

required_score  Numeric: A minimum of interaction score for an interaction to be included in the image. If not supplied, the threshold will be applied by STRING Based in the network. (Low Confidence = 150, Medium Confidence = 400, High Confidence = 700, Highest confidence = 900)

background  character vector: A set of STRING protein IDs to be used as the background proteome. Only STRING IDs are acceptable. (See rba_string_map_ids to map your IDs.)

...  rbioapi option(s). See rba_options’s arguments manual for more information on available options.

Value

A list with protein-protein interaction enrichment results.

Corresponding API Resources

"POST https://string-db.org/api/{output_format}/ppi_enrichment?identifiers={your_identifiers}&{optional_parameters}"

References

- STRING API Documentation
- Citations note on STRING website
**rba_string_homology_inter**

**See Also**

- rba_string_map_ids
- Other "STRING": rba_string_annotations(), rba_string_enrichment(), rba_string_homology_inter(), rba_string_homology_intra(), rba_string_interaction_partners(), rba_string_interactions_network(), rba_string_map_ids(), rba_string_network_image(), rba_string_version()

**Examples**

```r
rba_string_enrichment_ppi(ids = c("p53", "BRCA1", "cdk2", "Q99835", "CDC42", "CDK1", "KIF23", "PLK1", "RAC2", "RACGAP1"), species = 9606)
```

---

**rba_string_homology_inter**

*Get Similarity Scores Hits of Proteins in Different Species*

**Description**

Using this function, you can retrieve highest Smith-Waterman bit scores among your input proteins and proteins in every other STRING species (e.g. the closest homologous protein of your input protein in other species). Bit Scores serve as similarity scores between protein sequence; And, according to STRING documentations, as a proxy for protein homology.

**Usage**

```r
rba_string_homology_inter(ids, species = NULL, species_b = NULL, ...)
```

**Arguments**

- **ids**
  - Your protein ID(s). It is strongly recommended to supply STRING IDs. See rba_string_map_ids for more information.

- **species**
  - Numeric: NCBI Taxonomy identifier of your input proteins; Human Taxonomy ID is 9606. (Recommended, but optional if your input is less than 100 IDs.)

- **species_b**
  - (optional) Numeric: one or more NCBI Taxonomy identifiers of species to limit the closest homologous proteins search.

- **...**
  - rbioapi option(s). See rba_options’s arguments manual for more information on available options.

**Details**

Note that this function will return the highest similarity score hits of your given protein(s) and their closest homologous proteins in other species. To retrieve similarity scores of different proteins within the same species see rba_string_homology_intra.

Similarity matrix is imported -by STRING- from: *Similarity Matrix of Proteins (SIMAP)*
Value

A data frame with Your input proteins and it's closest homologous proteins among all other (or a defined) STRING species.

Corresponding API Resources

"POST https://string-db.org/api/{output-format}/homology_best? identifiers={your_identifiers}"

References

- STRING API Documentation
- Citations note on STRING website

See Also

rba_string_map_ids, rba_string_homology_intra

Other "STRING": rba_string_annotations(), rba_string_enrichment(), rba_string_enrichment_ppi(), rba_string_homology_intra(), rba_string_interaction_partners(), rba_string_interactions_network(), rba_string_map_ids(), rba_string_network_image(), rba_string_version()

Examples

```r
rba_string_homology_inter(ids = "p53",
        species = 9606,
        species_b = 7070)

rba_string_homology_inter(ids = "ENSP00000269305", species = 9606)
```

Description

Using this function, you can retrieve the Smith-Waterman bit scores among proteins of the same species. Bit Scores serve as similarity scores between protein sequence; And, according to STRING documentations, as a proxy for protein homology.
Usage

rba_string_homology_intra(ids, species = NULL, ...)

Arguments

ids        Your protein ID(s). It is strongly recommended to supply STRING IDs. See rba_string_map_ids for more information.
species   Numeric: NCBI Taxonomy identifier; Human Taxonomy ID is 9606. (Recommended, but optional if your input is less than 100 IDs.)
...        rbioapi option(s). See rba_options’s arguments manual for more information on available options.

Details

Note that this function will retrieve similarity scores of different proteins "within the same species". To Get a similarity scores of a given protein and it’s closets homologous proteins in other species, see rba_string_homology_inter.
Similarity matrix is imported -by STRING- from: Similarity Matrix of Proteins (SIMAP)

Value

A data frame with bit scores between your supplied proteins and their self-hit. To Reduce the transferred data, STRING returns only one half of the similarity matrix; This will not pose a problem because similarity matrix is symmetrical.

Corresponding API Resources

"POST https://string-db.org/api/{output-format}/homology?identifiers= {your_identifiers}"

References

• STRING API Documentation
• Citations note on STRING website

See Also

rba_string_map_ids, rba_string_homology_inter

Other "STRING": rba_string_annotations(), rba_string_enrichment(), rba_string_enrichment_ppi(), rba_string_homology_inter(), rba_string_interaction_partners(), rba_string_interactions_network(), rba_string_map_ids(), rba_string_network_image(), rba_string_version()
Examples

rba_string_homology_intra(ids = c("CDK1", "CDK2"), species = 9606)

rba_string_interactions_network

Get STRING Network Interactions

Description

This function will retrieve Sting interaction pairs among your input protein ids, with the combined score and separate score for each STRING score channels. You can further expand your network to a defined size by providing "add_node" parameter.

Usage

rba_string_interactions_network(
  ids,
  species = NULL,
  required_score = NULL,
  add_nodes = NULL,
  network_type = "functional",
  use_query_labels = FALSE,
  ...
)

Arguments

ids
  Your protein IDs. It is strongly recommended to supply STRING IDs. See rba_string_map_ids for more information.

species
  Numeric: NCBI Taxonomy identifier; Human Taxonomy ID is 9606. (Recommended, but optional if your input is less than 100 IDs.)

required_score
  Numeric: A minimum of interaction score for an interaction to be included in the image. If not supplied, the threshold will be applied by STRING Based in the network. (Low Confidence = 150, Medium Confidence = 400, High Confidence = 700, Highest confidence = 900)

add_nodes
  Numeric: Number of neighboring proteins to be added to the network. If none supplied by the user, this argument value will depend on the number of supplied "ids" argument:
  1. Single id: add_node will be set to 10 to retrieve the interaction neighborhood of your input protein.
  2. Multiple ids: add_node will be set to 0, thus the output will be the interactions between your input proteins.
network_type should be one of:

- "functional": (default) The edge’s indicate both physical and functional associations.
- "physical": The edges indicate that two proteins have a physical interaction or are parts of a complex.

use_query_labels

Logical: (Default = FALSE) Use the names supplied with the 'ids' argument as the nodes labels instead of STRING’s default ones.

rbioapi option(s). See rba_options’s arguments manual for more information on available options.

Details

Note that this function will return interactions between your set of supplied proteins, or at most, expand the interaction network by the given parameters. To retrieve a list of all possible interacting proteins with your given input, see rba_string_interaction_partners.

Value

A data frame which each row is a network interaction and the columns contains interactor information and interaction scores:

- stringId_A: STRING identifier (protein A)
- stringId_B:STRING identifier (protein B)
- preferredName_A: common protein name (protein A)
- preferredName_B: common protein name (protein B)
- ncbiTaxonId: NCBI taxon identifier
- score: combined score
- nscore: gene neighborhood score
- fscore: gene fusion score
- pscore: phylogenetic profile score
- ascore: co-expression score
- escore: experimental score
- dscore: database score
- tscore: textmining score

Corresponding API Resources

"POST https://string-db.org/api/{output-format}/network?identifiers= {your_identifiers}&{optional_parameters}"
References

- STRING API Documentation
- Citations note on STRING website

See Also

rba_string_map_ids, rba_string_interaction_partners

Other "STRING": rba_string_annotations(), rba_string_enrichment(), rba_string_enrichment_ppi(), rba_string_homology_inter(), rba_string_homology_intra(), rba_string_interaction_partners(), rba_string_map_ids(), rba_string_network_image(), rba_string_version()

Examples

```r
rba_string_interactions_network(ids = c("9606.ENSP00000269305", "9606.ENSP00000398698", "9606.ENSP00000275493"),
                                   network_type = "functional")
```

```r
rba_string_interactions_network(ids = c("9606.ENSP00000269305", "9606.ENSP00000398698", "9606.ENSP00000275493"),
                                   species = 9606,
                                   add_nodes = 10)
```

---

**rba_string_interaction_partners**

*Get All STRING Interaction Partners*

**Description**

This function will retrieve all the STRING interactions which include your proteins as one party of the interaction. (e.g. interaction between your proteins and every other STRING proteins.)

Given the size of STRING database, this function could return a very long results. See "Arguments" section for information on how to filter the interactions.
**Usage**

```r
rba_string_interaction_partners(
  ids,
  species = NULL,
  required_score = NULL,
  network_type = "functional",
  limit = NULL,
  ...
)
```

**Arguments**

- `ids` (Your protein ID(s)). It is strongly recommended to supply STRING IDs. See `rba_string_map_ids` for more information.

- `species` (Numeric: NCBI Taxonomy identifier; Human Taxonomy ID is 9606. Recommended, but optional if your input is less than 100 IDs.)

- `required_score` (Numeric: A minimum of interaction score for an interaction to be included in the image. If not supplied, the threshold will be applied by STRING Based in the network. (low Confidence = 150, Medium Confidence = 400, High Confidence = 700, Highest confidence = 900)

- `network_type` should be one of:
  - "functional": (default) The edge’s indicate both physical and functional associations.

- `limit` (Limit the number returned interaction partners per each of your input proteins. (e.g. Number of the most confident interaction partner to return per each input protein.)

- `...` (rbioapi option(s). See `rba_options`’s arguments manual for more information on available options.

**Details**

Note that this function will retrieve the interactions between your input proteins and every other STRING proteins. To retrieve the interaction among your input protein-set, see `rba_string_interactions_network`.

**Value**

A data frame which each row is a network interaction and the columns contains interactor information and interaction scores.

**Corresponding API Resources**

"POST https://string-db.org/api/{output-format}/interaction_partners?identifiers={your_identifiers}&{optional_parameters}"
References


- STRING API Documentation
- Citations note on STRING website

See Also

rba_string_map_ids, rba_string_interactions_network

Other "STRING": rba_string_annotations(), rba_string_enrichment(), rba_string_enrichment_ppi(), rba_string_homology_inter(), rba_string_homology_intra(), rba_string_interactions_network(), rba_string_map_ids(), rba_string_network_image(), rba_string_version()

Examples

rba_string_interaction_partners(ids = c("9606.ENSP00000269305", "9606.ENSP00000398698", "9606.ENSP00000275493"), network_type = "functional")

rba_string_interaction_partners(ids = "9606.ENSP00000269305", species = 9606, required_score = 700)

rba_string_map_ids

Map a Set of Identifiers to STRING Identifiers

Description

This function Calls STRING’s API to Convert a set of identifiers to STRING Identifiers. Although You can call STRING services with a variety of common identifiers, It is recommended by STRING’s documentations that you first map Your Protein/genes IDs to STRING IDs and then proceed with other STRING’s functions.

Usage

rba_string_map_ids(ids, species = NULL, echo_query = FALSE, limit = NULL, ...)
Arguments

ids           Your Common gene/protein Identifier(s) to be mapped.
species      Numeric: NCBI Taxonomy identifier; Human Taxonomy ID is 9606. (Recommended, but optional if your input is less than 100 IDs.)
echo_query   (default = FALSE) Include your input IDs as a column of the results.
limit        (Numeric, Optional) A limit on the number of matches per input ID. The output are sorted to have the best matches first.

...           rbioapi option(s). See rba_options’s arguments manual for more information on available options.

Value

A data frame with the mapped STRING IDs and other pertinent information.

Corresponding API Resources

"POST https://string-db.org/api/{output-format}/get_string_ids?identifiers= {your_identifiers}&{optional_parameters}"  

References

• STRING API Documentation
• Citations note on STRING website

See Also

Other "STRING": rba_string_annotations(), rba_string_enrichment(), rba_string_enrichment_ppi(), rba_string_homology_inter(), rba_string_homology_intra(), rba_string_interaction_partners(), rba_string_interactions_network(), rba_string_network_image(), rba_string_version()

Examples

rba_string_map_ids(ids = c("TP53", "TNF", "EGFR"), species = 9606)
rba_string_network_image

Get STRING Network Image

Description

Depending on that you supplied a single protein ID or more than one protein ID, this function will produce a static image of the interaction networks among your input proteins or/and with other proteins. See the "Arguments" section to learn more about how you can modify the network image.

Usage

rba_string_network_image(
  ids,
  image_format = "image",
  save_image = TRUE,
  species = NULL,
  add_color_nodes = NULL,
  add_white_nodes = NULL,
  required_score = NULL,
  network_flavor = "evidence",
  network_type = "functional",
  hide_node_labels = FALSE,
  use_query_labels = FALSE,
  hide_disconnected_nodes = FALSE,
  hide_structure_pics = FALSE,
  flat_nodes = FALSE,
  node_labels_center = FALSE,
  node_labels_font_size = 12,
  ...
)

Arguments

ids                   Your protein ID(s). It is strongly recommended to supply STRING IDs. See rba_string_map_ids for more information.
image_format          one of:
                       • "image": PNG image with normal resolution.
                       • "highres_image": High-resolution PNG image.
                       • "svg": Scalable Vector Graphics image.
save_image            Logical or Character:
                       • TRUE: Save the image to an automatically-generated path.
                       • FALSE: Do not save the image, just return it as an R object.
                       • Character string: A valid file path to save the image to.
species  Numeric: NCBI Taxonomy identifier; Human Taxonomy ID is 9606. (Recommended, but optional if your input is less than 100 IDs.)

add_color_nodes  Numeric: The number of colored nodes (queried proteins and first shell of interactors) to be added.

add_white_nodes  Numeric: The number of white nodes (second shell of interactors) to be added after colored nodes.

required_score  Numeric (Between 0 to 1000): A minimum of interaction score for an interaction to be included in the image. If not supplied, the threshold will be applied by STRING based on the network. (low Confidence = 150, Medium Confidence = 400, High Confidence = 700, Highest confidence = 900)

network_flavor  The style of network edges, should be one of:
  • "evidence": (default) Line’s color is based on the type of evidences that support the interaction.
  • "confidence": Line’s thickness is an indicator of the interaction’s confidence score.
  • "action": Line’s Shape is an indicator of the interaction’s predicted mode of actions.

network_type  should be one of:
  • "functional": (default) The edge’s indicate both physical and functional associations.
  • "physical": The edges indicate that two proteins have a physical interaction or are parts of a complex.

hide_node_labels  Logical: (Default = FALSE) Hide proteins names from the image

use_query_labels  Logical: (Default = FALSE) Use the names supplied with the 'ids' argument as the nodes labels instead of STRING’s default ones.

hide_disconnected_nodes  Logical: (Default = FALSE) Hide proteins that are not connected to any other proteins from the image

hide_structure_pics  Logical: (Default = FALSE) Hide protein’s structure picture from inside the bubbles

flat_nodes  Logical: (Default = FALSE) Make the nodes design flat instead of the default 3D design

node_labels_center  Logical: (Default = FALSE) Position the protein names labels center aligned on the nodes

node_labels_font_size  Numeric (Between 5 to 50, Default = 12) Font size of the protein nodes labels

...  
rbioapi option(s). See rba_options’s arguments manual for more information on available options.
Value

A network images which can be PNG or SVG depending on the inputs.

Corresponding API Resources

"POST https://string-db.org/api/{output-format}/network?identifiers= {your_identifiers}&{optional_parameters}"

References


• STRING API Documentation
• Citations note on STRING website

See Also

rba_string_map_ids

Other "STRING": rba_string_annotations(), rba_string_enrichment(), rba_string_enrichment_ppi(), rba_string_homology_inter(), rba_string_homology_intra(), rba_string_interaction_partners(), rba_string_interactions_network(), rba_string_map_ids(), rba_string_version()

Examples

## Not run:
```r
rba_string_network_image(ids = c("9606.ENSP00000269305", "9606.ENSP00000398698", "9606.ENSP00000275493"),
                         network_type = "functional",
                         save_image = FALSE)
```
## End(Not run)

## Not run:
```r
rba_string_network_image(ids = c("TP53", "TNF", "EGFR"),
                         species = 9606,
                         save_image = TRUE)
```
## End(Not run)

## Not run:
```r
rba_string_network_image(ids = "9606.ENSP00000269305",
                         image_format = "highres_image",
                         save_image = file.path(getwd(), "TP53_network.png"))
```
## End(Not run)
Description

Get STRING version and stable Address that this package currently uses.

Usage

rba_string_version(...)

Arguments

... rbioapi option(s). See rba_options’s arguments manual for more information on available options.

Details

Note that STRING releases new version at approximately 2 years cycle. Nevertheless, to insure reproducibility, STRING dedicates a stable address for each release. Thus you can always reproduce research and results obtained via a certain STRING version. If the version that rbioapi returns is outdated, kindly contact me.

Value

A list with STRING version and stable address.

Corresponding API Resources

"GET https://string-db.org/api/{output_format}/version"

References

• STRING API Documentation
• Citations note on STRING website

See Also

Other "STRING": rba_string_annotations(), rba_string_enrichment(), rba_string_enrichment_ppi(), rba_string_homology_inter(), rba_string_homology_intra(), rba_string_interaction_partners(), rba_string_interactions_network(), rba_string_map_ids(), rba_string_network_image()
rba_uniprot_antigens

Description

UniProt maps Antigenic features from different sources to the proteins' sequences. Using this function, you can retrieve all the Antigenic features that has been map to a given UniProt protein's sequence.

Usage

rba_uniprot_antigens(accession, ...)

Arguments

accession  
UniProtKB primary or secondary accession(s).

...  
rbioapi option(s). See rba_options's arguments manual for more information on available options.

Value

A list containing the Antigenic features of your supplied UniProt protein's sequence.

Corresponding API Resources

"GET https://www.ebi.ac.uk/proteins/api/antigen/{accession}" 

References

- Proteins API Documentation
- Citations note on UniProt website

See Also

Other "UniProt - Antigen": rba_uniprot_antigens_search()
**Examples**

```r
rba_uniprot_antigens("P04626")
```

**Description**

UniProt maps Antigenic (Antibody-binding) features from different sources to the proteins’ sequences. Using this function, you can search for Antigenic sequences that has been map to UniProt proteins. You may also refine your search with modifiers such as score etc. See "Arguments section" for more information.

**Usage**

```r
rba_uniprot_antigens_search(
    accession = NULL,
    antigen_sequence = NULL,
    antigen_id = NULL,
    ensembl_id = NULL,
    match_score = NULL,
    ...
)
```

**Arguments**

- `accession` **UniProtKB primary or secondary accession(s).** You can supply up to 100 accession numbers.
- `antigen_sequence` Protein sequence in the antigenic site.
- `antigen_id` Human Protein Atlas (HPA) antigen ID. You can supply up to 20 IDs.
- `ensembl_id` Ensembl Stable Transcript ID. You can supply up to 20 IDs.
- `match_score` (Numeric) Minimum alignment score for the antigen sequence and the target protein sequence.
- `...` `rbioapi` option(s). See `rba_options`'s arguments manual for more information on available options.

**Details**

Note that this is a search function. Thus, you are not required to fill every argument; You may use whatever combinations of arguments you see fit for your query.
**Value**

A list where each element correspond to a UniProt protein (search hit) and Antigenic features are organized under the "features" sub-list.

**Corresponding API Resources**

"GET https://www.ebi.ac.uk/proteins/api/antigen"

**References**

- Proteins API Documentation
- Citations note on UniProt website

**See Also**

Other "UniProt - Antigen": `rba_uniprot_antigens()`

**Examples**

```r
rba_uniprot_antigens_search(antigen_id = "HPA001060")
```

---

**rba_uniprot_coordinates**

*Get Genomic Coordinates of a Protein*

**Description**

Using this function you can retrieve genomic Coordinates of a Protein by either providing the protein’s UniProt accession or it’s ID in a cross-reference database (Ensembl, CCDC, HGNC or RefSeq). You should supply either ‘accession’ alone or ‘db_type’ and ‘db_id’ together.

**Usage**

```r
rba_uniprot_coordinates(accession = NULL, db_type = NULL, db_id = NULL, ...)
```
Arguments

accession  UniProtKB primary or secondary accession.
db_type   cross-reference database name, Should be one of: "Ensembl", "CCDC", "HGNC" or "RefSeq".
db_id     Protein's ID in the cross-reference database
...        rbioapi option(s). See rba_options's arguments manual for more information on available options.

Details


Value

A list with genome coordinates of your supplied protein.

Corresponding API Resources

"GET https://ebi.ac.uk/proteins/api/coordinates/{accession}"  
"GET https://ebi.ac.uk/proteins/api/coordinates/{dbtype}:{dbid}"

References

• Proteins API Documentation
• Citations note on UniProt website

See Also

Other "UniProt - Coordinates": rba_uniprot_coordinates_location(), rba_uniprot_coordinates_search(), rba_uniprot_coordinates_sequence()

Examples

rba_uniprot_coordinates(accession = "P25942")

rba_uniprot_coordinates(db_type = "HGNC", db_id = "CD40")
rba_uniprot_coordinates_location

Search UniProt entries by taxonomy and genomic coordinates

Description

For more information about how UniProt imports and calculates genomic coordinates data, see:

Usage

rba_uniprot_coordinates_location(
  taxid,
  locations,
  in_range = TRUE,
  feature = FALSE,
  ...
)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>taxid</td>
<td>NIH-NCBI Taxon ID.</td>
</tr>
<tr>
<td>locations</td>
<td>genomic location formatted as: chromosome:start-end. (e.g. &quot;Y:17100001-19600000&quot;). If you omit chromosome, it will be interpreted as any chromosome (e.g. &quot;1-10000&quot;).</td>
</tr>
<tr>
<td>in_range</td>
<td>Only return proteins that are in range.</td>
</tr>
<tr>
<td>feature</td>
<td>(logical) Get features?</td>
</tr>
<tr>
<td>...</td>
<td>rbioapi option(s). See rba_options’s arguments manual for more information on available options.</td>
</tr>
</tbody>
</table>

Value

a list containing UniProt proteins which match the supplied genomic location and taxonomy ID.

Corresponding API Resources

"GET https://ebi.ac.uk/proteins/api/coordinates/{taxonomy}/{locations}/feature"
"GET https://ebi.ac.uk/proteins/api/coordinates/{taxonomy}/{locations}"

References

rba_uniprot_coordinates_search

- Proteins API Documentation
- Citations note on UniProt website

See Also

Other "UniProt - Coordinates": rba_uniprot_coordinates(), rba_uniprot_coordinates_search(), rba_uniprot_coordinates_sequence()

Examples

rba_uniprot_coordinates_location(taxid = 9606,
locations = "Y:17100001-19600000", in_range = TRUE)

rba_uniprot_coordinates_location(taxid = 9606,
locations = "20:39000001", in_range = FALSE)

rba_uniprot_coordinates_search

Search Genomic Coordinates of UniProt entries

Description

Use this function to search genomic coordinates of UniProt entries. You may also refine your search with modifiers such as chromosome, taxon id etc. See "Arguments section" for more information.

Usage

rba_uniprot_coordinates_search(
  accession = NULL,
  chromosome = NULL,
  ensembl_id = NULL,
  gene = NULL,
  protein = NULL,
  taxid = NULL,
  location = NULL,
  ...
)
Arguments

accession     UniProtKB primary or secondary accession(s). You can supply up to 100 accession numbers.
chromosome    chromosome name, such as "X", "Y", 1, 20, etc. You can supply up to 20 values.
ensembl_id    Ensembl Stable gene ID, transcript ID or translation ID. You can supply up to 20 IDs.
gene          UniProt gene name(s). You can supply up to 20 gene names.
protein       UniProt protein name
taxid         NIH-NCBI Taxon ID. You can supply up to 20 taxon IDs.
location      Genome location range such as "58205437-58219305"
...           rbioapi option(s). See rba_options’s arguments manual for more information on available options.

Details

Note that this is a search function. Thus, you are not required to fill every argument; You may use whatever combinations of arguments you see fit for your query. For more information about how UniProt imports and calculates genomic coordinates data, see: McGarvey, P. B., Nightingale, A., Luo, J., Huang, H., Martin, M. J., Wu, C., & UniProt Consortium (2019). UniProt genomic mapping for deciphering functional effects of missense variants. Human mutation, 40(6), 694–705. https://doi.org/10.1002/humu.23738

Value

List where each element corresponds to one UniProt entity returned by your search query. The element itself is a sub-list containing that protein’s coordinates information.

Corresponding API Resources

"GET https://ebi.ac.uk/proteins/api/coordinates"

References

- Proteins API Documentation
- Citations note on UniProt website

See Also

Other "UniProt - Coordinates": rba_uniprot_coordinates(), rba_uniprot_coordinates_location(), rba_uniprot_coordinates_sequence()
Examples

```r
rba_uniprot_coordinates_search(taxid = 9606, chromosome = "y")
```

Description

Using this function you can retrieve genome coordinates of a given UniProt protein by providing protein position or position range. You can either supply 'p_position' alone or supply 'p_start' and 'p_end' together.

Usage

```r
rba_uniprot_coordinates_sequence(
  accession,
  p_position = NULL,
  p_start = NULL,
  p_end = NULL,
  ...
)
```

Arguments

- `accession`: UniProtKB primary or secondary accession.
- `p_position`: (numeric) Protein sequence position
- `p_start`: (numeric) Protein sequence position start
- `p_end`: (numeric) Protein sequence position end
- `...`: rbioapi option(s). See `rba_options`'s arguments manual for more information on available options.

Details


Value

Genome coordinates of your supplied proteins.
rba_uniprot_features

Get UniProt protein sequence features by accession

Description

Use this function to retrieve sequence annotations (features) of a protein by its UniProt accession.

Usage

rba_uniprot_features(accession, types = NULL, categories = NULL, ...)

Corresponding API Resources

"GET https://ebi.ac.uk/proteins/api/coordinates/location /{accession}:{pPosition}"  
"GET https://ebi.ac.uk/proteins/api/coordinates/location /{accession}:pStart-{pEnd}"

References


• Proteins API Documentation

• Citations note on UniProt website

See Also

Other "UniProt - Coordinates": rba_uniprot_coordinates(), rba_uniprot_coordinates_location(), rba_uniprot_coordinates_search()
Arguments

accession | UniProtKB primary or secondary accession.
categories| Sequence annotation (Features) categories (subsection). accepted values are: "MOLECULE_PROCESSING", "TOPOLOGY", "SEQUENCE_INFORMATION", "STRUCTURAL", "DOMAINS_AND_SITES", "PTM", "VARIANTS" and/or "MUTAGENESIS". You can supply up to 8 categories.

...rbioapi option(s). See rba_options’s arguments manual for more information on available options.

Value

A list in which you can find all of your given protein’s sequence annotations in a sub-list named "features".

Corresponding API Resources

"GET https://www.ebi.ac.uk/proteins/api/features/{accession}"

References

- Proteins API Documentation
- Citations note on UniProt website

See Also

Other "UniProt - Features": rba_uniprot_features_search()

Examples

```r
rba_uniprot_features("Q99616")
```
```r
rba_uniprot_features(accession = "Q99616", types = "DISULFID")
```
rba_uniprot_features_search

*UniProt maintains [R](https://www.uniprot.org/help/sequence_annotation) sequence annotations (features) that describe regions in the protein sequence. Using this function, you can search and retrieve UniProt proteins’ sequence annotations (features). You may also refine your search query with variety of modifiers.*

**Description**

Note that this is a search function. Thus, you are not required to fill every argument; You may use whatever combinations of arguments you see fit for your query. UniProt Entries are grouped in two sections:

1. Reviewed(Swiss-Prot): Manually annotated records with information extracted from literature and curator-evaluated computational analysis.
2. Unreviewed (TrEMBL): Computationally analyzed records that await full manual annotation.

**Usage**

```r
rba_uniprot_features_search(
    accession = NULL,
    gene = NULL,
    exact_gene = NULL,
    protein = NULL,
    reviewed = NULL,
    organism = NULL,
    taxid = NULL,
    categories = NULL,
    types = NULL,
    ...
)
```

**Arguments**

- `accession` *UniProtKB primary or secondary accession(s). You can supply up to 100 accession numbers.*
- `gene` *UniProt gene name(s). You can supply up to 20 gene names. e.g. if you supply "CD40", "CD40 ligand" will also be included.*
- `exact_gene` *UniProt exact gene name(s). You can supply up to 20 exact gene names. e.g. if you supply "CD40", "CD40 ligand" will not be included in the results.*
- `protein` *UniProt protein name*
reviewed Logical: If TRUE, only return "UniProtKB/Swiss-Prot" (reviewed) entries; If FALSE, only return TrEMBL (un-reviewed) entries.

organism Organism name.

taxid NIH-NCBI Taxon ID. You can supply up to 20 taxon IDs.

categories Sequence annotation (Features) categories (subsection). accepted values are: "MOLECULE_PROCESSING", "TOPOLOGY", "SEQUENCE_INFORMATION", "STRUCTURAL", "DOMAINS_AND_SITES", "PTM", "VARIANTS" and/or "MUTAGENESIS". You can supply up to 8 categories.


rbioapi option(s). See rba_options’s arguments manual for more information on available options.

Value

List where each element corresponds to one UniProt entity returned by your search query. The element itself is a sub-list containing all information that UniProt has about that entity.

Corresponding API Resources

"GET https://www.ebi.ac.uk/proteins/api/features"

References

- Proteins API Documentation
- Citations note on UniProt website

See Also

Other "UniProt - Features": rba_uniprot_features()
rba_uniprot_genecentric

Examples

rba_uniprot_features_search(accession = "Q99616")

rba_uniprot_features_search(gene = "cd40")

rba_uniprot_features_search(gene = "cd40 ligand")

rba_uniprot_features_search(gene = "cd40", reviewed = TRUE)

rba_uniprot_features_search(accession = "Q99616", categories = c("MOLECULE_PROCESSING", "TOPOLOGY"))

rba_uniprot_features_search(accession = "Q99616", types = "DISULFID")

rba_uniprot_genecentric

Get Gene-Centric proteins by UniProt Accession

Description

Using this function you can retrieve gene-centric data. For more information, see What are proteomes? and Automatic gene-centric isoform mapping for eukaryotic reference proteome entries.

Usage

rba_uniprot_genecentric(accession, ...)

Arguments

accession UniProtKB primary or secondary accession.
... rbioapi option(s). See rba_options’s arguments manual for more information on available options.

Value

A list containing information of Gene-Centric proteins.

Corresponding API Resources

"GET https://ebi.ac.uk/proteins/api/genecentric/{accession}"
rba_uniprot_genecentric_search

References


• Proteins API Documentation
• Citations note on UniProt website

See Also

Other "UniProt - Proteomes": rba_uniprot_genecentric_search(), rba_uniprot_proteomes(), rba_uniprot_proteomes_search()

Examples

rba_uniprot_genecentric("P29965")

rba_uniprot_genecentric_search

Search Gene-Centric Proteins

Description

Using this function you can search UniProt for available gene-centrics from proteomes. For more information, see What are proteomes? and Automatic gene-centric isoform mapping for eukaryotic reference proteome entries. You may also refine your search with modifiers upid, accession and gene. See ”Arguments section” for more information.

Usage

rba_uniprot_genecentric_search(upid = NULL, accession = NULL, gene = NULL, ...)

Arguments

upid UniProt Proteome identifier (UPID). You can supply up to 100 UPIDs.
accession UniProtKB primary or secondary accession(s). You can supply up to 100 accession numbers.
gene unique gene identifier(s) found in MOD, Ensembl, Ensembl Genomes, OLN, ORF or UniProt Gene Name.
... rbioapi option(s). See rba_options’s arguments manual for more information on available options.
rba_uniprot_genecentric_search

Details

Note that this is a search function. Thus, you are not required to fill every argument; You may use whatever combinations of arguments you see fit for your query.

Value

a list containing gene-centric proteins search hits.

Corresponding API Resources

"GET https://ebi.ac.uk/proteins/api/genecentric"

References


• Proteins API Documentation

• Citations note on UniProt website

See Also

Other "UniProt - Proteomes": rba_uniprot_genecentric(), rba_uniprot_proteomes(), rba_uniprot_proteomes_search()

Examples

rba_uniprot_genecentric_search(accession = "P59594")

rba_uniprot_genecentric_search(gene = "Spike")

rba_uniprot_genecentric_search(upid = "UP000000354")
Description

UniProt describes the effects of mutations in proteins’ amino acid sequence on the biological properties of the protein, cell or the organism. Using this function, you can get the Mutagenesis description that has been mapped to a given UniProt protein.

Usage

```
rba_uniprot_mutagenesis(accession, location = NULL, ...)
```

Arguments

- `accession`: UniProtKB primary or secondary accession(s).
- `location`: A valid amino acid range (e.g. 10-25) within the sequence range of the given protein.
- `...`: rbioapi option(s). See `rba_options`’s arguments manual for more information on available options.

Value

A list containing the mutagenesis description of your supplied UniProt protein’s sequence.

Corresponding API Resources

"GET https://www.ebi.ac.uk/proteins/api/mutagenesis/{accession}"

References

- Proteins API Documentation
- Citations note on UniProt website

See Also

Other "UniProt - Mutagenesis": `rba_uniprot_mutagenesis_search()`
**rba_uniprot_mutagenesis_search**

**Examples**

```r
rba_uniprot_mutagenesis(accession = "P0DTC2", location = "300-400")
```

**Description**

UniProt describes the effects of mutations in proteins’ amino acid sequence on the biological properties of the protein, cell or the organism. Using this function, you can search for mutagenesis description in UniProt proteins. You may also refine your search. See "Arguments section" for more information.

**Usage**

```r
rba_uniprot_mutagenesis_search(
  accession = NULL,
  taxid = NULL,
  db_id = NULL,
  ...
)
```

**Arguments**

- **accession**: UniProtKB primary or secondary accession(s). You can supply up to 100 accession numbers.
- **taxid**: NIH-NCBI Taxon ID. You can supply up to 20 taxon IDs.
- **db_id**: The ID in a Cross-reference (external) database. You can supply up to 20 values.
- **...**: rbioapi option(s). See `rba_options`'s arguments manual for more information on available options.

**Details**

Note that this is a search function. Thus, you are not required to fill every argument; You may use whatever combinations of arguments you see fit for your query.

**Value**

A list Where each element correspond to a UniProt protein (search hit) and mutagenesis description are organized under the "features" sub-list.
rba_uniprot_proteins

Corresponding API Resources

"GET https://www.ebi.ac.uk/proteins/api/mutagenesis"

References

- Proteins API Documentation
- Citations note on UniProt website

See Also

Other "UniProt - Mutagenesis": rba_uniprot_mutagenesis()

Examples

```r
# search all mutations in COVID19 proteins
rba_uniprot_mutagenesis_search(taxid = 2697049)
```

---

rba_uniprot_proteins  Get UniProt entry by accession

Description

Use this function to retrieve a UniProt Entry by it’s UniProt accession. You can also use "isoform" or "interaction" arguments to retrieve isoforms or interactor proteins of that entry. Note that in one function call you can only set none or only one of "isoform" or "interaction" as TRUE, not both of them.

Usage

```r
rba_uniprot_proteins(accession, interaction = FALSE, isoforms = FALSE, ...)
```

Arguments

- **accession**: UniProtKB primary or secondary accession.
- **interaction**: Logical: (default = FALSE) Only retrieve interaction information of your supplied UniProt entity?
isoforms Logical: (default = FALSE) Only retrieve isoforms of your supplied UniProt entity?

... rbioapi option(s). See rba_options's arguments manual for more information on available options.

Value

A list that contains UniProt protein informations with your supplied accession.

Corresponding API Resources

"GET https://ebi.ac.uk/proteins/api/proteins/{accession}"  
"GET https://ebi.ac.uk/proteins/api/proteins/interaction/{accession}"  
"GET https://ebi.ac.uk/proteins/api/proteins/{accession}/isoforms"

References


• Proteins API Documentation

• Citations note on UniProt website

See Also

Other "UniProt - Proteins": rba_uniprot_proteins_crossref(), rba_uniprot_proteins_search()

Examples

rba_uniprot_proteins(accession = "P01730")

rba_uniprot_proteins(accession = "P01730", interaction = TRUE)

rba_uniprot_proteins(accession = "Q29983", isoforms = TRUE)
rba_uniprot_proteins_crossref

*Get UniProt Entry by UniProt Cross-Reference Database and ID*

**Description**

**UniProt Cross-Reference** links protein Entities with cross-reference (external) databases. Using this function, you can retrieve a UniProt entity using external database name and protein ID in that database.

**Usage**

```r
rba_uniprot_proteins_crossref(
    db_id,
    db_name,
    reviewed = NULL,
    isoform = NULL,
    ...
)
```

**Arguments**

- `db_id`  
The protein ID in the cross-reference (external) database.
- `db_name`  
cross-reference (external database) name.
- `reviewed`  
Logical: (Optional) If TRUE, only returns "UniProtKB/Swiss-Prot" (reviewed) entries; If FALSE, only returns TrEMBL (un-reviewed) entries.
- `isoform`  
Numeric: (Optional) you have two options:
  - 0: Exclude isoforms.
  - 1: Return isoforms only.

  see: Alternative products

- `...`  
rbioapi option(s). See `rba_options`’s arguments manual for more information on available options.

**Value**

List which each element is a UniProt entity that correspond to your supplied cross-reference database name and ID.

**Corresponding API Resources**

"GET https://www.ebi.ac.uk/proteins/api/proteins/{dbtype}:{dbid}"
rba_uniprot_proteins_search

References

- Proteins API Documentation
- Citations note on UniProt website

See Also

Other "UniProt - Proteins": rba_uniprot_proteins(), rba_uniprot_proteins_search()

Examples

rba_uniprot_proteins_crossref("cd40", "hgnc")

rba_uniprot_proteins_crossref("cd40", "hgnc", reviewed = TRUE)

rba_uniprot_proteins_crossref("mica", "hgnc", isoform = 0)

rba_uniprot_proteins_search

Search UniProt entries

Description

Using this function, you can search and retrieve UniProt Knowledge-base (UniProtKB) protein entries using variety of options. You may also refine your search with modifiers such as sequence length, review status etc. See "Arguments" section" for more information.

Usage

rba_uniprot_proteins_search(
    accession = NULL,
    reviewed = NULL,
    isoform = NULL,
    go_term = NULL,
    keyword = NULL,
    ec = NULL,
    gene = NULL,
Arguments

accession  UniProtKB primary or secondary accession(s). You can supply up to 100 accession numbers.

reviewed  Logical: If TRUE, only return "UniProtKB/Swiss-Prot" (reviewed) entries; If FALSE, only return TrEMBL (un-reviewed) entries.

isoform  Numeric: you have three options:
- 0: Exclude isoforms.
- 1: Return isoforms only.
- 2: Return both.

see: Alternative products

go_term  Limit the search to entries associated with your supplied GO (Gene Ontology) term. You can supply either GO ID or a character string - partially or fully-matching the term. e.g. "GO:0001776" or "leukocyte homeostasis". If you supply "leukocyte", any term containing that word will be included, e.g. "leukocyte chemotaxis", "leukocyte activation".

keyword  Limit the search to entries that contain your supplied keyword. see: UniProt Keywords

ec  EC (Enzyme Commission) number(s). You can supply up to 20 EC numbers.

gene  UniProt gene name(s). You can supply up to 20 gene names. e.g. if you supply "CD40", "CD40 ligand" will also be included.

exact_gene  UniProt exact gene name(s). You can supply up to 20 exact gene names. e.g. if you supply "CD40", "CD40 ligand" will not be included in the results.

protein  UniProt protein name

organism  Organism name.

taxid  NIH-NCBI Taxon ID. You can supply up to 20 taxon IDs.

pubmed  Entries which cite to the article with your supplied PubMed ID.

seq_length  An exact sequence length (e.g. 150) or a range of sequence lengths (e.g. "130-158").

md5  Sequence md5 value.

...  rbioapi option(s). See rba_options’s arguments manual for more information on available options.
Details

Note that this is a search function. Thus, you are not required to fill every argument; You may use whatever combinations of arguments you see fit for your query.

UniProt Entries are grouped in two sections:

1. Reviewed (Swiss-Prot): Manually annotated records with information extracted from literature and curator-evaluated computational analysis.
2. Unreviewed (TrEMBL): Computationally analyzed records that await full manual annotation.

Value

A List where each element corresponds to one UniProt entity returned by your search query. The element itself is a sub-list containing all information that UniProt has about that entity.

Corresponding API Resources

"GET https://www.ebi.ac.uk/proteins/api/proteins"

References

- Proteins API Documentation
- Citations note on UniProt website

See Also

Other "UniProt - Proteins": rba_uniprot_proteins(), rba_uniprot_proteins_crossref() 

Examples

rba_uniprot_proteins_search(accession = "Q99616")

rba_uniprot_proteins_search(gene = "cd40")

rba_uniprot_proteins_search(gene = "cd40 ligand")

rba_uniprot_proteins_search(gene = "cd40", reviewed = TRUE)

rba_uniprot_proteins_search(gene = "cd40", reviewed = TRUE, isoform = 1)
Description

UniProt collects and annotates proteomes (protein sets expressed in an organism). Using this function you can search UniProt for available proteomes. See What are proteomes? for more information.

Usage

`rba_uniprot_proteomes(upid, get_proteins = FALSE, reviewed = NULL, ...)`

Arguments

- **upid**: UniProt Proteome identifier (UPID). You can supply up to 100 UPIDs.
- **get_proteins**: logical: set FALSE (default) to only return information of the proteome with supplied UPID, set TRUE to also return the proteins of the supplied proteome UPID.
- **reviewed**: Logical: Only considered when get_proteins is TRUE. If TRUE, only return "UniProtKB/Swiss-Prot" (reviewed) proteins; If FALSE, only return TrEMBL (un-reviewed) entries. Leave it as NULL if you do not want to filter proteins based on their review status.
- **...**: rbioapi option(s). See rba_options's arguments manual for more information on available options.

Value

A list containing information of the proteome with your supplied UPID that can contain the proteomes protein entries based on the value of get_proteins argument.

Corresponding API Resources

"GET https://ebi.ac.uk/proteins/api/proteomes/proteins/{upid}"  
"GET https://ebi.ac.uk/proteins/api/proteomes/{upid}"
References


• Proteins API Documentation
• Citations note on UniProt website

See Also

Other "UniProt - Proteomes": `rba_uniprot_genecentric()`, `rba_uniprot_genecentric_search()`, `rba_uniprot_proteomes_search()`

Examples

```r
rba_uniprot_proteomes(upid = "UP000000354")

rba_uniprot_proteomes(upid = "UP000000354", get_proteins = TRUE)
```

---

**rba_uniprot_proteomes_search**

Search Proteomes in UniProt

---

Description

UniProt collects and annotates proteomes (Protein sets expressed in an organism). Using this function you can search UniProt for available proteomes. see What are proteomes? for more information. You may also refine your search with modifiers such as keyword, taxon id etc. See "Arguments section" for more information.

Usage

```r
rba_uniprot_proteomes_search(
  name = NULL,
  upid = NULL,
  taxid = NULL,
  keyword = NULL,
  xref = NULL,
  genome_acc = NULL,
  is_ref_proteome = NULL,
  is_redundant = NULL,
)```
Arguments

- **name**: a keyword in proteome’s name
- **upid**: UniProt Proteome identifier (UPID). You can supply up to 100 UPIDs.
- **taxid**: NIH-NCBI Taxon ID. You can supply up to 20 taxon IDs.
- **keyword**: Limit the search to entries that contain your supplied keyword. See: UniProt Keywords
- **xref**: Proteome cross-references such as Genome assembly ID or Biosample ID. You can supply up to 20 cross-reference IDs.
- **genome_acc**: Genome accession associated with the proteome’s components.
- **is_ref_proteome**: (logical) If TRUE, only return reference proteomes; If FALSE, only returns non-reference proteomes; If NULL (default), the results will not be filtered by this criteria see ‘What are reference proteomes?’ for more information.
- **is_redundant**: (logical) If TRUE, only return redundant proteomes; If FALSE, only returns non-redundant proteomes; If NULL (default), the results will not be filtered by redundancy. See ‘Reducing proteome redundancy’ for more information.
- **...**

Details

Note that this is a search function. Thus, you are not required to fill every argument; You may use whatever combinations of arguments you see fit for your query.

Value

A list where each element is a list that corresponds to a single proteome (search hit) and contains informations pertinent to that proteome.

Corresponding API Resources

"GET https://ebi.ac.uk/proteins/api/proteomes"

References

- Proteins API Documentation
- Citations note on UniProt website
rba_uniprot_proteomics

See Also

Other "UniProt - Proteomes": rba_uniprot_genecentric(), rba_uniprot_genecentric_search(), rba_uniprot_proteomes()

Examples

rba_uniprot_proteomes_search(name = "SARS-CoV")

rba_uniprot_proteomes_search(name = "SARS-CoV", is_ref_proteome = TRUE)

rba_uniprot_proteomes_search(name = "SARS-CoV", is_ref_proteome = TRUE)

rba_uniprot_proteomes_search(genome_acc = "AY274119")

rba_uniprot_proteomics

Get Proteomics Peptides Mapped to UniProt Protein

Description

UniProt maps proteomics peptides from different sources to the proteins’ sequences. Using this function, you can retrieve all the proteomics peptides features that has been map to a given UniProt protein’s sequence.

Usage

rba_uniprot_proteomics(accession, ...)

Arguments

accession UniProtKB primary or secondary accession.
...
rbioapi option(s). See rba_options’s arguments manual for more information on available options.

Value

A list containing the proteomics peptides features of your supplied UniProt protein’s sequence.

Corresponding API Resources

"GET https://www.ebi.ac.uk/proteins/api/proteomics/{accession}"
rba_uniprot_proteomics_search

Description

UniProt maps proteomics peptides from different sources to the proteins’ sequences. Using this function, you can search for proteomics peptides that has been map to UniProt proteins. You may also refine your search with modifiers such as data_source, peptide etc. See "Arguments section" for more information.

Usage

```r
rba_uniprot_proteomics_search(
  accession = NULL,
  data_source = NULL,
  taxid = NULL,
  upid = NULL,
  peptide = NULL,
  unique = NULL,
  ...
)
```
Arguments

accession  
UniProtKB primary or secondary accession(s). You can supply up to 100 accession numbers.

data_source  
Proteomics data source. You can choose up to two of:
  - "MaxQB"
  - "PeptideAtlas"
  - "EPD"
  - "ProteomicsDB"

taxid  
NIH-NCBI Taxon ID. You can supply up to 20 taxon IDs.

upid  
UniProt Proteome identifier (UPID). You can supply up to 100 UPIDs.

peptide  
Peptide sequence(s). You can supply up to 20 sequences.

unique  
Logical: Should the results be filtered based on the Peptide’s uniqueness (the fact that a peptide maps to only 1 protein). If TRUE, Only unique peptides will be returned, if FALSE only un-unique peptides will be returned; If NULL (default) the results will not be filtered based on this.

...  
rbioapi option(s). See `rba_options`’s arguments manual for more information on available options.

Details

Note that this is a search function. Thus, you are not required to fill every argument; You may use whatever combinations of arguments you see fit for your query.

see also: Mass spectrometry-based proteomics data in UniProtKB

Value

A list Where each element correspond to a UniProt protein and proteomics peptides are organized under the "features" sub-list.

Corresponding API Resources

"GET https://www.ebi.ac.uk/proteins/api/proteomics"

References

- Proteins API Documentation
- Citations note on UniProt website
**rba_uniprot_ptm**

Get Post-Translational Modification of UniProt Protein

---

**Description**

UniProt maps post-translational modification features from different sources to the proteins’ sequences. Using this function, you can retrieve all the post-translational modification features that has been map to a given UniProt protein’s sequence.

**Usage**

```r
rba_uniprot_ptm(accession, ...)  
```

**Arguments**

- `accession` — UniProtKB primary or secondary accession.
- `...` — rbioapi option(s). See `rba_options`’s arguments manual for more information on available options.

**Details**

see also: PTM / Processing section in UniProtKB

**Value**

A list containing the post-translational modification features of your supplied UniProt protein’s sequence.

---

**See Also**

Other "UniProt - Proteomics": `rba_uniprot_proteomics()`

**Examples**

```r
rba_uniprot_proteomics_search(peptide = "MEDYTKIEK")

rba_uniprot_proteomics_search(peptide = "MEDYTKIEK")

## Not run:
### this will generate a very large response!
  rba_uniprot_proteomics_search(taxid = 9606,
               data_source = "PeptideAtlas",
               progress = TRUE, timeout = 999999, unique = TRUE)

## End(Not run)
```
Corresponding API Resources

"GET https://www.ebi.ac.uk/proteins/api/proteomics-ptm/{accession}"  

References

- Proteins API Documentation
- Citations note on UniProt website

See Also

Other "UniProt - PTM": rba_uniprot_ptm_search()

Examples

```
rba_uniprot_ptm(accession = "P04234")
```

Description

UniProt maps proteomics peptides from different sources to the proteins’ sequences. Using this function, you can search for proteomics peptides that has been map to UniProt proteins. You may also refine your search with modifiers such as data_source, peptide etc. See "Arguments section" for more information.

Usage

```
rba_uniprot_ptm_search(
    accession = NULL,
    ptm = NULL,
    data_source = NULL,
    taxid = NULL,
    upid = NULL,
    peptide = NULL,
    unique = NULL,
    ...
)
```
Arguments

accession  UniProtKB primary or secondary accession(s). You can supply up to 100 accession numbers.

ptm  Post-translational modification name

data_source  Proteomics data source. You can choose up to two of:
   - "MaxQB"
   - "PeptideAtlas"
   - "EPD"
   - "ProteomicsDB"

taxid  NIH-NCBI Taxon ID. You can supply up to 20 taxon IDs.

upid  UniProt Proteome identifier (UPID). You can supply up to 100 UPIDs.

peptide  Peptide sequence(s). You can supply up to 20 sequences.

unique  Logical: Should the results be filtered based on the Peptide’s uniqueness (the fact that a peptide maps to only 1 protein). If TRUE, Only unique peptides will be returned, if FALSE only un-unique peptides will be returned; If NULL (default) the results will not be filtered based on this.

...  rbioapi option(s). See rba_options’s arguments manual for more information on available options.

Details

Note that this is a search function. Thus, you are not required to fill every argument; You may use whatever combinations of arguments you see fit for your query.

see also: PTM / Processing section in UniProtKB

Value

A list Where each element correspond to a UniProt protein and post-translational modification are organized under the "features" sub-list.

Corresponding API Resources

"GET https://www.ebi.ac.uk/proteins/api/proteomics-ptm"

References

- Proteins API Documentation
- Citations note on UniProt website
See Also

Other "UniProt - PTM": \texttt{rba_uniprot_ptm()}

Examples

\begin{verbatim}
rba_uniprot_ptm_search(peptide = "NDQVYQPLRDRTDAQYSHLGGNWAR")
\end{verbatim}

---

**rba_uniprot_taxonomy** \textit{Get UniProt Taxonomy Nodes}

Description

Using this function, you can retrieve taxonomic nodes information by providing their NCBI taxonomic identifiers. Also, you can explicitly retrieve other nodes in relation to your supplied node’s hierarchy in UniProt Taxonomy database.

Usage

\begin{verbatim}
rba_uniprot_taxonomy(
  ids, 
  hierarchy = NULL, 
  node_only = TRUE, 
  page_size = 200, 
  page_number = 1, 
  ...
)
\end{verbatim}

Arguments

- **ids** (numeric) a single or a numeric vector of NCBI taxonomic identifier(s)
- **hierarchy** Retrieve taxonomic nodes that have specific hierarchical relation to your supplied taxonomic node. Should be one of: "children", "parent" or "siblings".
- **node_only** Retrieve only the node(s) information and exclude URL links to parents, siblings and children nodes.
- **page_size** (numeric) Only when hierarchy is supplied. Hierarchy information may be very long, thus UniProt API will paginate the results, you may use this argument to control the pagination. Maximum value is 200.
- **page_number** (numeric) Only when hierarchy is supplied. Hierarchy information may be very long, thus UniProt API will paginate the results, you may use this argument to control the pagination.
- \texttt{...} \texttt{rbioapi} option(s). See \texttt{rba_options}’s arguments manual for more information on available options.
Value

a list containing your supplied nodes or their related nodes taxonomic information.

Corresponding API Resources

"GET https://ebi.ac.uk/proteins/api/ids/{ids}"  
"GET https://ebi.ac.uk/proteins/api/ids/{id}/node"  
"GET https://ebi.ac.uk/proteins/api/id/{id}/node"  
"GET https://ebi.ac.uk/proteins/api/id/{id}/children"  
"GET https://ebi.ac.uk/proteins/api/id/{id}/children/node"  
"GET https://ebi.ac.uk/proteins/api/id/{id}/parent"  
"GET https://ebi.ac.uk/proteins/api/id/{id}/parent/node"  
"GET https://ebi.ac.uk/proteins/api/id/{id}/siblings"  
"GET https://ebi.ac.uk/proteins/api/id/{id}/siblings/node"

References


• Proteins API Documentation

• Citations note on UniProt website

See Also

Other "UniProt - Taxonomy": rba_uniprot_taxonomy_lca(), rba_uniprot_taxonomy_lineage(), rba_uniprot_taxonomy_name(), rba_uniprot_taxonomy_path(), rba_uniprot_taxonomy_relationship()

Examples

rba_uniprot_taxonomy(ids = c(9606, 10090))

rba_uniprot_taxonomy(ids = 9989, hierarchy = "children")
rba_uniprot_taxonomy_lca

Get Lowest Common Ancestor (LCA) of Two Taxonomy Nodes

Description

Use this function to retrieve lowest common ancestor (LCA) of two taxonomy nodes in UniProt Taxonomy database

Usage

rba_uniprot_taxonomy_lca(ids, ...)

Arguments

ids (numeric) Numeric vector of NCBI taxonomic identifiers, with minimum length of two.

... rbioapi option(s). See rba_options’s arguments manual for more information on available options.

Value

A list with UniProt taxonomy information of your supplied taxonomy elements.

Corresponding API Resources

"GET https://ebi.ac.uk/proteins/api/ancestor/{ids}"

References


• Proteins API Documentation

• Citations note on UniProt website

See Also

Other "UniProt - Taxonomy": rba_uniprot_taxonomy(), rba_uniprot_taxonomy_lineage(), rba_uniprot_taxonomy_name(), rba_uniprot_taxonomy_path(), rba_uniprot_taxonomy_relationship()
**rba_uniprot_taxonomy_lineage**

**Examples**

```r
rba_uniprot_taxonomy_lca(c(9606,10090,9823,7712))
```

---

**rba_uniprot_taxonomy_lineage**

*Get Taxonomic Lineage*

**Description**

Use this function to retrieve the taxonomic lineage of your supplied taxonomy node.

**Usage**

```r
rba_uniprot_taxonomy_lineage(id, ...)
```

**Arguments**

- `id` *(numeric)* a NCBI taxonomic identifier
- `...` rbioapi option(s). See `rba_options`’s arguments manual for more information on available options.

**Value**

A list with a data frame containing All the nodes that preceded your supplied node in the taxonomic tree. with your node as the first row and the root node in the last row.

**Corresponding API Resources**

"GET https://ebi.ac.uk/proteins/api/lineage/{id}"

"GET https://ebi.ac.uk/proteins/api/lineage/{id}"

**References**

- Proteins API Documentation
- Citations note on UniProt website
See Also

Other "UniProt - Taxonomy": `rba_uniprot_taxonomy()`, `rba_uniprot_taxonomy_lca()`, `rba_uniprot_taxonomy_name()`, `rba_uniprot_taxonomy_path()`, `rba_uniprot_taxonomy_relationship()`

Examples

```r
rba_uniprot_taxonomy_lineage(id = 9989)
```

**Description**

Using this function, you can search and retrieve taxonomic nodes using their names from UniProt Taxonomy database.

**Usage**

```r
rba_uniprot_taxonomy_name(
  name,
  field = "scientific",
  search_type = "equal_to",
  node_only = TRUE,
  page_size = 200,
  page_number = 1,
  ...
)
```

**Arguments**

- `name` a name to to be used as search query.
- `field` Specify the field that your supplied name should be searched. It should be one of: "scientific" (default), "common" or "mnemonic".
- `search_type` The logical relationship between your supplied search query and the taxonomic name field. It should be one of "equal_to" (default), "start_with", "end_with" or "contain".
- `node_only` (logical) Retrieve only the node(s) information and exclude URL links to parents, siblings and children nodes. default = TRUE
- `page_size` (numeric) Your search results may be very long, thus UniProt API will paginate the results, you may use this argument to control the pagination. maximum value is 200.
rba_uniprot_taxonomy_name

page_number  (numeric) Your search results may be very long, thus UniProt API will paginate the results, you may use this argument to control the pagination. maximum value is 200.

rbiapi option(s). See rba_options’s arguments manual for more information on available options.

Value

a list containing taxonomic nodes that match your supplied inputs.

Corresponding API Resources

"GET https://ebi.ac.uk/proteins/api/name/{name}"  
"GET https://ebi.ac.uk/proteins/api/name/{name}/node"

References


• Proteins API Documentation
• Citations note on UniProt website

See Also

Other "UniProt - Taxonomy": rba_uniprot_taxonomy(), rba_uniprot_taxonomy_lca(), rba_uniprot_taxonomy_lineage(), rba_uniprot_taxonomy_path(), rba_uniprot_taxonomy_relationship()

Examples

rba_uniprot_taxonomy_name(name = "homo", field = "scientific", search_type = "start_with")

rba_uniprot_taxonomy_name(name = "adenovirus", field = "scientific", search_type = "contain", page_size = 200, page_number = 2)
rba_uniprot_taxonomy_path

Traverse UniProt Taxonomic Tree Path

Description

Using this function you can retrieve nodes that are located in the top or the bottom of your supplied node in UniProt Taxonomy database tree.

Usage

rba_uniprot_taxonomy_path(id, direction, depth = 5, ...)

Arguments

- id (numeric) a NCBI taxonomic identifier
- direction direction of the taxonomic path, either "TOP" or "BOTTOM".
- depth (numeric) How many levels should be traversed on the taxonomic tree? (from 1 to 5, default = 5)
- ... rbioapi option(s). See rba_options’s arguments manual for more information on available options.

Value

A nested list containing the node which are in the path specified by your supplied argument in the UniProt taxonomic tree.

Corresponding API Resources

"GET https://ebi.ac.uk/proteins/api/path"

References

- Proteins API Documentation
- Citations note on UniProt website

See Also

Other "UniProt - Taxonomy": rba_uniprot_taxonomy(), rba_uniprot_taxonomy_lca(), rba_uniprot_taxonomy_lineage(), rba_uniprot_taxonomy_name(), rba_uniprot_taxonomy_relationship()
Examples

rba_uniprot_taxonomy_path(id = 9606, direction = "TOP", depth = 3)

rba_uniprot_taxonomy_path(id = 207598, direction = "BOTTOM", depth = 3)

Description

Use this function to retrieve the shortest path between two nodes in the taxonomy tree of UniProt Taxonomy database.

Usage

rba_uniprot_taxonomy_relationship(from, to, ...)

Arguments

from         NCBI taxonomic identifier of your initial node.
to           NCBI taxonomic identifier of your final node.
...          rbioapi option(s). See rba_options’s arguments manual for more information on available options.

Value

a nested list containing the node which are in the shortest path between your supplied nodes.

Corresponding API Resources

"GET https://ebi.ac.uk/proteins/api/relationship"

References

- Proteins API Documentation
- Citations note on UniProt website
rba_uniprot_uniparc

See Also

Other "UniProt - Taxonomy": rba_uniprot_taxonomy(), rba_uniprot_taxonomy_lca(), rba_uniprot_taxonomy_lineage(), rba_uniprotTaxonomy_name(), rba_uniprotTaxonomy_path()

Examples

rba_uniprotTaxonomy_relationship(from = 9606, to = 10090)

rba_uniprot_uniparc  Get UniParc entry

Description

Use this function to retrieve UniParc entries. You can use either - and only one of - UniProt accession, Cross-reference database id, UniParc ID or UniProt Proteome UPID. You can also filter the returned content of the returned UniParc entry. see "Argument" section for more details.

Usage

rba_uniprot_uniparc(
  upi = NULL,
  accession = NULL,
  db_id = NULL,
  upid = NULL,
  rf_dd_type = NULL,
  rf_db_id = NULL,
  rf_active = NULL,
  rf_tax_id = NULL,
  ...
)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>upi</td>
<td>unique UniParc Identifier.</td>
</tr>
<tr>
<td>accession</td>
<td>UniProtKB primary or secondary accession.</td>
</tr>
<tr>
<td>db_id</td>
<td>Protein ID in the cross-reference (external) database.</td>
</tr>
<tr>
<td>upid</td>
<td>UniProt Proteome identifier (UPID). You can supply up to 100 UPIDs.</td>
</tr>
<tr>
<td>rf_dd_type</td>
<td>Filter the content of the UniParc entry by cross-reference names. You can supply multiple values.</td>
</tr>
<tr>
<td>rf_db_id</td>
<td>Filter the content of the UniParc entry by protein identifiers in any cross-reference database. You can supply multiple values.</td>
</tr>
<tr>
<td>rf_active</td>
<td>(logical) Filter the content of UniParc entry based on active status on source database:</td>
</tr>
</tbody>
</table>
rba_uniprot_uniparc

• NULL: (default) don’t filter contents based on active status.
• TRUE: only return contents which are still active.
• FALSE: Only return contents which are not active.

rf_tax_id (Numeric) Filter the content of the UniParc entry by NIH-NCBI Taxon ID. You can supply multiple values.

... rbioapi option(s). See rba_options’s arguments manual for more information on available options.

Value

A list which correspond to a UniParc entry.

Corresponding API Resources

"GET https://ebi.ac.uk/proteins/api/uniparc/accession/{accession}" 
"GET https://ebi.ac.uk/proteins/api/uniparc/dbreference/{dbid}" 
"GET https://ebi.ac.uk/proteins/api/uniparc/proteome/{upid}" 
"GET https://ebi.ac.uk/proteins/api/uniparc/upi/{upi}"

References

• Proteins API Documentation
• Citations note on UniProt website

See Also

Other "UniProt - UniParc": rba_uniprot_uniparc_bestguess(), rba_uniprot_uniparc_search(), rba_uniprot_uniparc_sequence()

Examples

rba_uniprot_uniparc(upi = "UPI00000000C9")

rba_uniprot_uniparc(upi = "UPI00000000C9")

rba_uniprot_uniparc(upi = "UPI00000000C9", rf_active = FALSE)
rba_uniprot_uniparc_bestguess

*Get UniParc Longest Sequence for Entries*

**Description**

This function returns the UniParc Entry with a cross-reference to the longest active UniProtKB sequence (preferably from Swiss-Prot and if not then TrEMBL). If it finds more than one longest active UniProtKB sequence it returns 400 (Bad Request) error response with the list of cross-references found.

**Usage**

```r
rba_uniprot_uniparc_bestguess(
  upi = NULL,
  accession = NULL,
  db_id = NULL,
  gene = NULL,
  taxid = NULL,
  ...
)
```

**Arguments**

- `upi` unique UniParc Identifier.
- `accession` UniProtKB primary or secondary accession(s). You can supply up to 100 accession numbers.
- `db_id` Protein ID in the cross-reference (external) database. You can supply up to 100 IDs.
- `gene` UniProt gene name(s). You can supply up to 20 gene names.
- `taxid` NIH-NCBI Taxon ID. You can supply up to 20 taxon IDs.
- `...` rbioapi option(s). See `rba_options`'s arguments manual for more information on available options.

**Value**

A list where each element correspond to a UniParc entry.

**Corresponding API Resources**

"GET https://ebi.ac.uk/proteins/api/uniparc/bestguess"
References

- Proteins API Documentation
- Citations note on UniProt website

See Also

Other "UniProt - UniParc": rba_uniprot_uniparc(), rba_uniprot_uniparc_search(), rba_uniprot_uniparc_sequence()

Examples

```r
rba_uniprot_uniparc_bestguess("UPI00000000C9")
```

**rba_uniprot_uniparc_search**

*Search UniParc Entries*

**Description**

Use this function to search UniProt Archive (UniParc) entries. You may also refine your search with modifiers such as sequence length, taxon id etc. See "Arguments section" for more information.

**Usage**

```r
rba_uniprot_uniparc_search(
  upi = NULL,
  accession = NULL,
  db_type = NULL,
  db_id = NULL,
  gene = NULL,
  protein = NULL,
  taxid = NULL,
  organism = NULL,
  sequence_checksum = NULL,
  ipr = NULL,
  signature_db = NULL,
  signature_id = NULL,
  upid = NULL,
)```
Arguments

upi unique UniParc Identifier(s). You can supply up to 100 IDs.
accession UniProtKB primary or secondary accession(s). You can supply up to 100 accession numbers.
db_type cross-reference (external database) name.
db_id Protein ID in the cross-reference (external) database. You can supply up to 100 IDs.
gene UniProt gene name(s). You can supply up to 20 gene names.
protein UniProt protein name.
taxid NIH-NCBI Taxon ID. You can supply up to 20 taxon IDs.
organism Organism name.
sequence_checksum Sequence CRC64 checksum.
pr InterPro identifier(s). You can supply up to 20 IDs.
signature_db InterPro’s signature database. You can supply up to 13 of the following values: “CATH”, “CDD”, “HAMAP”, “MobiDB Lite”, “Panther”, “Pfam”, “PIRSF”, “PRINTS”, “Prosite”, “SFLD”, “SMART”, “SUPERFAMILY” and/or “TIGR-fams”
signature_id Signature ID in the InterPro’s signature database. You can supply up to 20 IDs.
upid UniProt Proteome identifier (UPIID). You can supply up to 100 UPIDs.
seq_length An exact sequence length (e.g. 150) or a range of sequence lengths (e.g. “130-158”).
rf_dd_type Filter the content of the each UniParc entry by cross-reference names. You can supply multiple values.
rf_db_id Filter the content of the each UniParc entry by protein identifiers in any cross-reference database. You can supply multiple values.
rf_active (logical ) Filter the content of each UniParc entry based on active status on source database:
  • NULL: (default) don’t filter contents based on active status.
  • TRUE: only return contents which are still active.
  • FALSE: Only return contents which are not active.
rf_tax_id (Numeric) Filter the content of each UniParc entry by NIH-NCBI Taxon ID. You can supply multiple values.
... rbioapi option(s). See rba_options’s arguments manual for more information on available options.
Details

Note that this is a search function. Thus, you are not required to fill every argument; You may use whatever combinations of arguments you see fit for your query.

Value

A List where each element corresponds to one UniParc entry returned by your search query. The element itself is a sub-list containing sequence information and reference entries.

Corresponding API Resources

"GET https://ebi.ac.uk/proteins/api/uniparc"

References


• Proteins API Documentation

• Citations note on UniProt website

See Also

Other "UniProt - UniParc": rba_uniprot_uniparc(), rba_uniprot_uniparc_bestguess(), rba_uniprot_uniparc_sequence()

Examples

rba_uniprot_uniparc_search(upi = "UPI00000000C9")

rba_uniprot_uniparc_search(accession = "P30914")

rba_uniprot_uniparc_search(accession = "P30914", rf_active = TRUE)

rba_uniprot_uniparc_search(taxid = "694009", protein = "Nucleoprotein")
rba_uniprot_uniparc_sequence

Get UniParc Entries by Sequence

Description

Retrieve UniParc Entry by providing an exact sequence. Note that partial matches will not be accepted. You can also filter the returned content of the returned UniParc entry. see "Argument" section for more details.

Usage

```
rba_uniprot_uniparc_sequence(
    sequence,
    rf_dd_type = NULL,
    rf_db_id = NULL,
    rf_active = NULL,
    rf_tax_id = NULL,
    ...
)
```

Arguments

- **sequence**
  - Exact UniParc protein sequence. Partial matches will not be accepted.
- **rf_dd_type**
  - Filter the content of the UniParc entry by cross-reference names. You can supply multiple values.
- **rf_db_id**
  - Filter the content of the UniParc entry by protein identifiers in any cross-reference database. You can supply multiple values.
- **rf_active**
  - (logical) Filter the content of UniParc entry based on active status on source database:
    - NULL: (default) don’t filter contents based on active status.
    - TRUE: only return contents which are still active.
    - FALSE: Only return contents which are not active.
- **rf_tax_id**
  - (Numeric) Filter the content of the UniParc entry by NIH-NCBI Taxon ID. You can supply multiple values.
- **...**
  - rbioapi option(s). See rba_options’s arguments manual for more information on available options.

Value

A list which correspond to a UniParc entry.

Corresponding API Resources

"POST https://ebi.ac.uk/proteins/api/uniparc/sequence"
rba_uniprot_variation

References


• Proteins API Documentation

• Citations note on UniProt website

See Also

Other "UniProt - UniParc": rba_uniprot_uniparc(), rba_uniprot_uniparc_bestguess(), rba_uniprot_uniparc_search()

Examples

rba_uniprot_uniparc_sequence("GMRSCPRCSQGRCEGRCVCNPGYTEDC")

rba_uniprot_variation

Get natural variants in UniProt by NIH-NCBI SNP database identifier

Description

Retrieve natural variant annotations of a sequence using UniProt protein accession, dbSNP or HGVS expression.

Usage

rba_uniprot_variation(
    id,
    id_type,
    source_type = NULL,
    consequence_type = NULL,
    wild_type = NULL,
    alternative_sequence = NULL,
    location = NULL,
    save_peff = FALSE,
    ...
)
Arguments

id An ID which can be either a UniProt primary or secondary accession, NIH-NCBI dbSNP ID or HGVS expression. NIH-NCBI dbSNP id or HGVS Expression.

id_type The type of supplied ID argument, one of: "uniprot", "dbsnp" or "hgvs"

source_type Variation’s source type. You can choose up to two of: "UniProt", "large scale study" and/or "mixed".

consequence_type Variation’s consequence type. You can choose up to two of: "missense", "stop gained" or "stop lost".

wild_type Wild type amino acid. Accepted values are IUPAC single-letter amino acid (e.g. D for Aspartic acid) and "*" for stop codon. You can supply up to 20 values.

alternative_sequence Alternative amino acid. Accepted values are IUPAC single-letter amino acid (e.g. D for Aspartic acid) and "*" for stop codon and "-" for deletion. You can supply up to 20 values.

location A valid amino acid range (e.g. 10-25) within the sequence range where the variation occurs. You can supply up to 20 values.

save_peff Logical or Character:

• FALSE: (default) Do not save PEFF file, just return as a list object.
• TRUE: Save as PEFF file to an automatically-generated path.
• Character string: A valid file path to save the PEFF file.

... rbioapi option(s). See rba_options’s arguments manual for more information on available options.

Value

A list where each element is a list that corresponds to a UniProt protein entry.

Corresponding API Resources

"GET https://www.ebi.ac.uk/proteins/api/variation/dbsnp/{dbid}"  
"GET https://www.ebi.ac.uk/proteins/api/variation/hgvs/{hgvs}"  
"GET https://www.ebi.ac.uk/proteins/api/variation/{accession}"  

References


• Proteins API Documentation

• Citations note on UniProt website
**rba_uniprot_variation_search**

See Also

Other "UniProt - Variation": rba_uniprot_variation_search()

Examples

```r
rba_uniprot_variation(id = "rs121434451", id_type = "dbsnp")

rba_uniprot_variation(id = "NC_000008.11:g.22119227C>T", id_type = "hgvs")

rba_uniprot_variation(id = "O43593", id_type = "uniprot")
```

---

**rba_uniprot_variation_search**

Search UniProt Natural Variants

---

**Description**

Using this function, you can search and retrieve Natural variant(s) that has been annotated in the protein’s sequences. You may also refine your search with modifiers such as source type, disease etc. See "Arguments section" for more information.

**Usage**

```r
rba_uniprot_variation_search(
  accession = NULL,
  source_type = NULL,
  consequence_type = NULL,
  wild_type = NULL,
  alternative_sequence = NULL,
  location = NULL,
  disease = NULL,
  omim = NULL,
  evidence = NULL,
  taxid = NULL,
  db_type = NULL,
  db_id = NULL,
  save_peff = FALSE,
  ...
)
```
Arguments

accession    UniProtKB primary or secondary accession(s). You can supply up to 100 accession numbers.
source_type  Variation’s source type. You can choose up to two of: "UniProt", "large scale study" and/or "mixed".
consequence_type  Variation’s consequence type. You can choose up to two of: "missense", "stop gained" or "stop lost".
wild_type  Wild type amino acid. Accepted values are IUPAC single-letter amino acid (e.g. D for Aspartic acid) and "*" for stop codon. You can supply up to 20 values.
alternative_sequence  Alternative amino acid. Accepted values are IUPAC single-letter amino acid (e.g. D for Aspartic acid) and "*" for stop codon and "-" for deletion. You can supply up to 20 values.
location  A valid amino acid range (e.g. 10-25) within the sequence range where the variation occurs. You can supply up to 20 values.
disease  Human disease that are associated with a sequence variation. Accepted values are disease name (e.g. Alzheimer disease 18), partial disease name (Alzheimer) and/or disease acronym (e.g. AD). You can supply up to 20 values.
omim  OMIM ID that is associated with a variation. You can supply up to 20 values.
evidence  Pubmed ID of the variation’s citation. You can supply up to 20 values.
taxid  NIH-NCBI Taxon ID. You can supply up to 20 taxon IDs.
db_type  cross-reference database of the variation. You can supply up to two of the following:
  • "dbSNP": NIH-NCBI dbSNP database.
  • "cosmic curate": COSMIC (the Catalogue of Somatic Mutations in Cancer)
  • "ClinVar": NIH-NCBI ClinVar
db_id  The variation ID in a Cross-reference (external) database. You can supply up to 20 values.
save_peff  Logical or Character:
  • FALSE: (default) Do not save PEFF file, just return as a list object.
  • TRUE: Save as PEFF file to an automatically-generated path.
  • Character string: A valid file path to save the PEFF file.
...  rbioapi option(s). See rba_options’s arguments manual for more information on available options.

Details

Note that this is a search function. Thus, you are not required to fill every argument; You may use whatever combinations of arguments you see fit for your query.

Value

List where each element corresponds to one UniProt entity returned by your search query. The element itself is a sub-list containing all information that UniProt has about that Variation.
Corresponding API Resources

"GET https://www.ebi.ac.uk/proteins/api/variation"

References

- Proteins API Documentation
- Citations note on UniProt website

See Also

Other "UniProt - Variation": rba_uniprot_variation()

Examples

```r
rba_uniprot_variation_search(accession = "P05067")

rba_uniprot_variation_search(disease = "alzheimer disease, 18")

rba_uniprot_variation_search(disease = "alzheimer",
   wild_type = "A", alternative_sequence = "T")
```
Index

* Helper
  rba_connection_test, 4
  rba_options, 46
  rba_pages, 48

rba_connection_test, 4, 47, 48
rba_enrichr, 5, 8–10, 12–14
rba_enrichr_add_list, 6, 7, 7, 9, 10, 12–14
rba_enrichr_enrich, 6–8, 9, 12–14
rba_enrichr_gene_map, 7, 8, 10, 11, 13, 14
rba_enrichr_libs, 5–10, 12, 14
rba_enrichr_view_list, 7, 8, 10, 12, 13
rba_jaspar_collections, 15, 17, 19–23, 25, 26, 28–30, 32, 33
rba_jaspar_collections_matrices, 15, 16, 19, 21–23, 25, 26, 28–30, 32, 33
rba_jaspar_matrix, 15, 17, 18, 21–23, 25, 26, 28–30, 32, 33
rba_jaspar_matrix_search, 15, 17, 19, 22, 23, 25, 26, 28–30, 32, 33
rba_jaspar_matrix_versions, 15, 17, 19, 21, 21, 23, 25, 26, 28–30, 32, 33
rba_jaspar_releases, 15, 17, 19, 21, 22, 23, 25, 26, 28–30, 32, 33
rba_jaspar_sites, 15, 17, 19, 21–23, 24, 26, 28–30, 32, 33
rba_jaspar_species, 15, 17, 19–23, 25, 27–30, 32, 33
rba_jaspar_species_matrices, 15, 17, 19, 21–23, 25, 26, 29, 30, 32, 33
rba_jaspar_taxons, 15, 17, 19–23, 25, 26, 28, 29, 30, 32, 33
rba_jaspar_taxons_matrices, 15, 17, 19, 21–23, 25, 26, 29, 30, 32, 33
rba_jaspar_tffm, 15, 17, 19, 21–23, 25, 26, 28–30, 31, 33
rba_jaspar_tffm_search, 15, 17, 19, 21–23, 25, 26, 28–30, 32, 33
rba_mieaa_cats, 34, 36, 38–40, 42–45
rba_mieaa_convert_type, 35, 36, 38, 40, 42, 43, 45
rba_mieaa_convert_version, 35, 36, 37, 39, 40, 42–45
rba_mieaa_enrich, 35, 36, 38, 41–43, 45
rba_mieaa_enrich_results, 35, 36, 38, 40, 41, 42, 43, 45
rba_mieaa_enrich_status, 35, 36, 38, 40–42, 42, 45
rba_mieaa_enrich_submit, 35, 36, 38, 40, 42, 43
rba_pages, 5, 17, 20, 27, 30, 33, 47, 48
rba_panther_enrich, 49, 52–55, 57, 58
rba_panther_family, 50, 51, 53–55, 57, 58
rba_panther_homolog, 50, 52, 54, 55, 57, 58
rba_panther_info, 49–53, 53, 55–58
rba_panther_mapping, 50, 52–54, 55, 57, 58
rba_panther_ortholog, 50, 52–55, 56, 58
rba_panther_tree_grafter, 50, 52–55, 57, 58
rba_reactome_analysis, 59, 63, 65, 67–69, 71, 73, 82, 84, 87–90
rba_reactome_analysis_download, 61, 62, 64, 65, 67–69, 71–73
rba_reactome_analysis_import, 61, 63, 64, 67–69, 71–73

182
INDEX

rba_reactome_analysis_mapping, 61, 63, 65, 66, 69, 71, 73
rba_reactome_analysis_pdf, 61, 63, 65, 67, 67, 71, 73
rba_reactome_analysis_species, 61, 63, 65, 66, 67, 69, 73, 96
rba_reactome_analysis_token, 61, 63, 65, 67, 69, 71, 71
rba_reactome_complex_list, 73, 75, 78, 99
rba_reactome_complex_subunits, 74, 74, 78, 99
rba_reactome_diseases, 76
rba_reactome_entity_other_forms, 74, 75, 77, 99
rba_reactome_event_ancestors, 78, 81
rba_reactome_event_hierarchy, 79, 80
rba_reactome_exporter_diagram, 81, 85, 88–90
rba_reactome_exporter_event, 84, 84, 88, 90
rba_reactome_exporter_overview, 84, 85, 86, 90
rba_reactome_exporter_reaction, 83–85, 88, 88
rba_reactome_interactors_psicquic, 90, 93
rba_reactome_interactors_static, 91, 92
rba_reactome_mapping, 94
rba_reactome_orthology, 71, 95
rba_reactome_participant_of, 74, 75, 78, 98, 98
rba_reactome_participants, 97, 99, 102, 111
rba_reactome_pathways_events, 100, 102, 104
rba_reactome_pathways_low, 101, 101, 104
rba_reactome_pathways_top, 101, 102, 103
rba_reactome_people_id, 104, 107
rba_reactome_people_name, 105, 106
rba_reactome_query, 107
rba_reactome_species, 60, 67, 69, 71, 80, 86, 93–95, 102, 103, 108
rba_reactome_version, 110
rba_reactome_xref, 111
rba_string_annotations, 112, 114, 115, 117–119, 122, 124, 125, 128, 129
rba_string_enrichment, 113, 114, 117–119, 122, 124, 125, 128, 129
rba_string_enrichment_ppi, 113, 115, 115, 118, 119, 122, 124, 125, 128, 129
rba_string_homology_inter, 113, 115, 117, 119, 122, 124, 125, 128, 129
rba_string_homology_intra, 113, 115, 117, 118, 119, 122, 124, 125, 128, 129
rba_string_interaction_partners, 113, 115, 117–119, 121, 122, 122, 125, 128, 129
rba_string_interactions_network, 113, 115, 117–119, 120, 123–125, 128, 129
rba_string_map_ids, 112–120, 122–124, 124, 126, 128, 129
rba_string_network_image, 113, 115, 117–119, 122, 124, 125, 126, 129
rba_string_version, 113, 115, 117–119, 122, 124, 125, 128, 129
rba_uniprot_antigens, 130, 132
rba_uniprot_antigens_search, 130, 131
rba_uniprot_coordinates, 132, 135, 136, 138
rba_uniprot_coordinates_location, 133, 134, 136, 138
rba_uniprot_coordinates_search, 133, 135, 135, 138
rba_uniprot_coordinates_sequence, 133, 135, 136, 137
rba_uniprot_features, 138, 141
rba_uniprot_features_search, 139, 140
rba_uniprot_gene-centric, 142, 144, 154, 156
rba_uniprot_gene-centric_search, 143, 143, 154, 156
rba_uniprot_mutagenesis, 145, 147
rba_uniprot_mutagenesis_search, 145, 146
rba_uniprot_proteins, 147, 150, 152
rba_uniprot_proteins_crossref, 148, 149, 152
rba_uniprot_proteins_search, 148, 150, 150
rba_uniprot_proteomes, 143, 144, 153, 156
rba_uniprot_proteomes_search, 143, 144, 154, 154
rba_uniprot_proteomics, 156, 159
rba_uniprot_proteomics_search, 157, 157
rba_uniprot_pseudo, 159, 162
rba_uniprot_ptm_search, 160, 160
rba_uniprot_taxonomy, 162, 164, 166–168, 170
rba_uniprot_taxonomy_lca, 163, 164, 166–168, 170
rba_uniprot_taxonomy_lineage, 163, 164, 165, 167, 168, 170
rba_uniprot_taxonomy_name, 163, 164, 166, 166, 168, 170
rba_uniprot_taxonomy_path, 163, 164, 166, 167, 168, 170
rba_uniprot_taxonomy_relationship, 163, 164, 166–168, 169
rba_uniprot_uniparc, 170, 173, 175, 177
rba_uniprot_uniparc_bestguess, 171, 172, 175, 177
rba_uniprot_uniparc_search, 171, 173, 173, 177
rba_uniprot_uniparc_sequence, 171, 173, 175, 176
rba_uniprot_variation, 177, 181
rba_uniprot_variation_search, 179, 179