Package ‘rscopus’

Type Package
Title Scopus Database 'API' Interface
Version 0.6.3
Date 2018-11-08
Maintainer John Muschelli <muschellij2@gmail.com>
Description Uses Elsevier 'Scopus' API
   <https://dev.elsevier.com/sc_apis.html> to download
   information about authors and their citations.
License GPL-2
Depends R (>= 3.2.0)
Imports httr, utils, stats, plyr, tidyr, dplyr, tools
LazyData TRUE
Suggests xml2, rvest, graphics, testthat, jpeg
URL https://dev.elsevier.com/sc_apis.html,
     https://github.com/muschellij2/rscopus
BugReports https://github.com/muschellij2/rscopus/issues
RoxygenNote 6.1.0
Encoding UTF-8
NeedsCompilation no
Author John Muschelli [aut, cre]
Repository CRAN
Date/Publication 2018-11-19 21:50:03 UTC

R topics documented:

   abstract_retrieval ........................................... 3
   affiliation_retrieval .................................... 4
   affil_df .................................................... 5
   affil_list_to_df ......................................... 6
<table>
<thead>
<tr>
<th>R topics documented:</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>affil_search</td>
<td>6</td>
</tr>
<tr>
<td>all_possible_affils</td>
<td>7</td>
</tr>
<tr>
<td>article_retrieval</td>
<td>7</td>
</tr>
<tr>
<td>author_df</td>
<td>8</td>
</tr>
<tr>
<td>author_retrieval_id</td>
<td>10</td>
</tr>
<tr>
<td>author_search</td>
<td>11</td>
</tr>
<tr>
<td>author_search_by_affil</td>
<td>12</td>
</tr>
<tr>
<td>citation_retrieval</td>
<td>13</td>
</tr>
<tr>
<td>collapse_affil</td>
<td>14</td>
</tr>
<tr>
<td>complete_multi_author_info</td>
<td>14</td>
</tr>
<tr>
<td>elsevier_authenticate</td>
<td>15</td>
</tr>
<tr>
<td>embase_retrieval</td>
<td>15</td>
</tr>
<tr>
<td>entitlement_retrieval</td>
<td>16</td>
</tr>
<tr>
<td>entries_to_affil_list</td>
<td>17</td>
</tr>
<tr>
<td>entries_to_citation_df</td>
<td>17</td>
</tr>
<tr>
<td>entries_to_df</td>
<td>18</td>
</tr>
<tr>
<td>entry_to_affil</td>
<td>18</td>
</tr>
<tr>
<td>generic_elsevier_api</td>
<td>19</td>
</tr>
<tr>
<td>gen_entries_to_df</td>
<td>20</td>
</tr>
<tr>
<td>get_affiliation_info</td>
<td>20</td>
</tr>
<tr>
<td>get_api_key</td>
<td>21</td>
</tr>
<tr>
<td>get_author_info</td>
<td>22</td>
</tr>
<tr>
<td>get_complete_author_info</td>
<td>22</td>
</tr>
<tr>
<td>get_links</td>
<td>23</td>
</tr>
<tr>
<td>inst_token_header</td>
<td>24</td>
</tr>
<tr>
<td>metadata_retrieval</td>
<td>25</td>
</tr>
<tr>
<td>multi_author_info</td>
<td>26</td>
</tr>
<tr>
<td>nonull</td>
<td>26</td>
</tr>
<tr>
<td>object_retrieval</td>
<td>27</td>
</tr>
<tr>
<td>print.scopus_api_key</td>
<td>28</td>
</tr>
<tr>
<td>process_affiliation_name</td>
<td>29</td>
</tr>
<tr>
<td>process_author_name</td>
<td>29</td>
</tr>
<tr>
<td>read_eto</td>
<td>30</td>
</tr>
<tr>
<td>recommendation_retrieval</td>
<td>30</td>
</tr>
<tr>
<td>replace_non_ascii</td>
<td>31</td>
</tr>
<tr>
<td>scopus_search</td>
<td>32</td>
</tr>
<tr>
<td>set_api_key</td>
<td>33</td>
</tr>
<tr>
<td>subject_areas</td>
<td>33</td>
</tr>
</tbody>
</table>

Index                                                                  | 34   |
abstract_retrieval

SCOPUS Abstract Retrieval

Description

This function wraps generic_elsevier_api to give a retrieval of an abstract from the Elsevier Abstract Retrieval API.

Usage

abstract_retrieval(id, identifier = c("scopus_id", "eid", "doi", "pii", "pubmed_id", "pui", "group_id"), http_end = NULL, ...)

Arguments

   id                Identifier for abstract
   identifier       Type of identifier to use
   http_end         any additional end to http statement. See generic_elsevier_api
   ...              Arguments to be passed to generic_elsevier_api

Value

List of elements, similar to generic_elsevier_api

See Also

generic_elsevier_api

Examples

api_key = get_api_key(NULL, error = FALSE)
if (!is.null(api_key)){
   x = abstract_retrieval("51053811915002700", identifier = "pii",
   verbose = FALSE)
}
affiliation_retrieval  SCOPUS Affiliation Retrieval

Description

This function wraps generic_elsevier_api to give a retrieval of an affiliation from the Elsevier Affiliation Retrieval API

Usage

affiliation_retrieval(id, identifier = c("affiliation_id", "eid"),
http_end = NULL, ...)

Arguments

id  Identifier for affiliation
identifier  Type of identifier to use
http_end  any additional end to http statement. See generic_elsevier_api
...  Arguments to be passed to generic_elsevier_api

Value

List of elements, similar to generic_elsevier_api

See Also

generic_elsevier_api

Examples

api_key = get_api_key(NULL, error = FALSE)
if (!is.null(api_key)){
  x = affiliation_retrieval("60006183", identifier = "affiliation_id",
    verbose = FALSE)
}
affil_df

Search Affiliation Content on SCOPUS

Description

Searches SCOPUS to get information about documents on an affiliation

Usage

affil_df(affil_id = NULL, affil_name = NULL, api_key = NULL, verbose = TRUE, facets = NULL, sort = "document-count", ...)

affil_data(affil_id = NULL, affil_name = NULL, api_key = NULL, verbose = TRUE, facets = NULL, sort = "document-count", ...)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>affil_id</td>
<td>Affiliation ID number.</td>
</tr>
<tr>
<td>affil_name</td>
<td>affiliation name. Disregarded if affil_id is specified</td>
</tr>
<tr>
<td>api_key</td>
<td>Elsevier API key</td>
</tr>
<tr>
<td>verbose</td>
<td>Print diagnostic messages</td>
</tr>
<tr>
<td>facets</td>
<td>Facets sent in query. See <a href="https://dev.elsevier.com/api_docs.html">https://dev.elsevier.com/api_docs.html</a></td>
</tr>
<tr>
<td>sort</td>
<td>sorting sent to query</td>
</tr>
<tr>
<td>...</td>
<td>Arguments to be passed to author_search</td>
</tr>
</tbody>
</table>

Value

List of entries from SCOPUS

Note

The affil_data command will return the list of all entries as well as the data.frame

See Also

get_author_info
affil_list_to_df  
List of SCOPUS Entries to List of Affiliations Data Frames

Description
Take a SCOPUS entry and transform it to a data frame of affiliations

Usage
affil_list_to_df(affils)

Arguments

affils  List of affiliations, from entries_to_affil_list

Value
A data.frame of affiliation information. A column named "index" denotes the element of the object affils that the row corresponds to

affil_search  
Search Author Content on SCOPUS

Description
Searches SCOPUS to get information about documents on an author.

Usage
affil_search(affil_id, searcher = "AF-ID",
http = "https://api.elsevier.com/content/search/affiliation",
facets = "affilcountry(sort=document-count)", count = 200, ...

Arguments

affil_id  Affiliation ID number
searcher  Identifier for Affiliation ID. Do not change unless you know exactly what the API calls for.
http  Address for scopus api
facets  Facets sent in query. See https://dev.elsevier.com/api_docs.html
count  number of records to retrieve (below 200, see https://dev.elsevier.com/api_key_settings.html)
...  Arguments to be passed to author_search
all_possible_affils

Value
List of entries from SCOPUS

See Also
get_author_info

Description
Take a SCOPUS entry and transform it to a data frame of all affiliations listed in there

Usage
all_possible_affils(entries)

Arguments
entries list of entries from SCOPUS, usually from author_search result

Value
A data.frame of affiliations

article_retrieval    ScienceDirect Article Retrieval

Description
This function wraps generic_elsevier_api to give a retrieval of an article from the Elsevier Article Retrieval API

Usage
article_retrieval(id, view = c("META", "META_ABS", "META_ABS_REF", "FULL", "REF", "ENTITLED"), identifier = c("scopus_id", "eid", "doi", "pii", "pubmed_id"), http_end = NULL, ...)
author_df

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Identifier for article</td>
</tr>
<tr>
<td>view</td>
<td>Which view to see. See <a href="https://dev.elsevier.com/guides/ArticleRetrievalViews.htm">https://dev.elsevier.com/guides/ArticleRetrievalViews.htm</a></td>
</tr>
<tr>
<td>identifier</td>
<td>Type of identifier to use</td>
</tr>
<tr>
<td>http_end</td>
<td>any additional end to http statement. See generic_elsevier_api</td>
</tr>
</tbody>
</table>

Value

List of elements, similar to generic_elsevier_api

See Also

generic_elsevier_api

Examples

```r
api_key = get_api_key(NULL, error = FALSE)
if (!is.null(api_key)){
x = article_retrieval("S1053811915002700", identifier = "pii",
verbose = FALSE, view = "FULL")
gen = x$content$'full-text-retrieval-response`
    ot = gen$originalText
} else {
x = article_retrieval("S1053811915002700",
    identifier = "pii",
    api_key_error = FALSE)
}
```

author_df

Search Author Content on SCOPUS

Description

Searches SCOPUS to get information about documents on an author. Note, author_list returns a list of the entries from author_search, but allows you to put in a name.

Usage

```r
author_df(au_id = NULL, last_name = NULL, first_name = NULL,
api_key = NULL, verbose = TRUE, all_author_info = FALSE,
http = "https://api.elsevier.com/content/search/scopus",
    view = "COMPLETE", count = 25, general = TRUE, scrub = FALSE,
    ...)
```

```r
author_df_orig(..., general = FALSE)
```
author_df

author_list(au_id = NULL, last_name = NULL, first_name = NULL, 
api_key = NULL, verbose = TRUE, 
http = "https://api.elsevier.com/content/search/scopus", 
view = "COMPLETE", count = 25, ...)

author_data(..., verbose = TRUE, all_author_info = FALSE, 
general = TRUE, scrub = FALSE)

Arguments

au_id                  Author ID number. Overrides any first/last name argument
last_name             last name of author
first_name             first name of author
api_key                Elsevier API key
verbose                Print diagnostic messages
all_author_info        Should all author info be recorded instead of that just to the author given
http                   Address for scopus api
view                   type of view to give, see https://api.elsevier.com/documentation/ScopusSearchAPI.wadl
count                  number of records to retrieve (below 25, see https://dev.elsevier.com/api_key_settings.html)
general                Should gen_entries_to_df instead of the way before version 0.5.10.9001
scrub                  Should 'scrub_identifier' be run on the identifier?
...                    Arguments to be passed to author_search

Value

List of entries from SCOPUS

Note

The author_data command will return the list of all entries as well as the data.frame.

See Also

get_author_info

Examples

if (have_api_key()) {
    res = author_df(last_name = "Muschelli", first_name = "John", 
    verbose = FALSE)
}
author_retrieval_id  SCOPUS Author Retrieval

Description
This function wraps generic_elsevier_api to give a retrieval of an author from the Elsevier Author Retrieval API.

Usage

author_retrieval_id(id, identifier = c("author_id", "eid"), http_end = NULL, ...)
multi_author_retrieval(id, identifier = c("author_id", "eid"), http_end = NULL, ...)
author_retrieval(au_id = NULL, last_name = NULL, first_name = NULL, view = c("LIGHT", "STANDARD", "ENHANCED", "METRICS", "ENTITLED"), self_cite = c("include", "exclude"), http_end = NULL, verbose = TRUE, api_key = NULL, ...)

Arguments

id  Identifier for author
identifier  Type of identifier to use
http_end  any additional end to http statement. See generic_elsevier_api
...  Arguments to be passed to generic_elsevier_api
au_id  Author ID number. Overrides any first/last name argument
last_name  last name of author
first_name  first name of author
view  Which view to see. See https://api.elsevier.com/documentation/AuthorRetrievalAPI.wadl
self_cite  Should self-citations be included?
verbose  Print diagnostic messages
api_key  Elsevier API key

Value
List of elements, similar to generic_elsevier_api

Note
See https://api.elsevier.com/documentation/AuthorRetrievalAPI.wadl for documentation.
author_search

See Also

generic_elsevier_api

Examples

api_key = get_api_key(NULL, error = FALSE)
if (!is.null(api_key)){
    x = author_retrieval(au_id = "40462056100",
                        verbose = FALSE)
    x = author_retrieval_id("40462056100", identifier = "author_id",
                          verbose = FALSE)
} else {
    x = author_retrieval_id("40462056100",
                            identifier = "author_id",
                            api_key_error = FALSE, verbose = FALSE)
    x = author_retrieval(au_id = "40462056100",
                        api_key_error = FALSE, verbose = FALSE)
}

author_search

Search Author Content on SCOPUS

Description

Searches SCOPUS to get information about documents on an author.

Usage

author_search(au_id, api_key = NULL,
               http = "https://api.elsevier.com/content/search/author", count = 200,
               start = 0, verbose = TRUE, facets = "subjarea(sort=fd,count=350)",
               searcher = "AU-ID", max_count = Inf, view = c("STANDARD",
                  "COMPLETE"), add_query = NULL, ...)

Arguments

au_id Author ID number
api_key API Key for Elsevier
http Address for scopus api
count number of records to retrieve (below 200, see https://dev.elsevier.com/api_key_settings.html)
start where should the records start gathering
verbose Print diagnostic messages
facets Facets sent in query. See https://dev.elsevier.com/api_docs.html
author_search_by_affil

<table>
<thead>
<tr>
<th>searcher</th>
<th>Identifier for author ID. Do not change unless you know exactly what the API calls for.</th>
</tr>
</thead>
<tbody>
<tr>
<td>max_count</td>
<td>Maximum count of records to be returned.</td>
</tr>
<tr>
<td>view</td>
<td>type of view to give, see <a href="https://api.elsevier.com/documentation/AuthorSearchAPI.wadl">https://api.elsevier.com/documentation/AuthorSearchAPI.wadl</a></td>
</tr>
<tr>
<td>add_query</td>
<td>Things to add to the query parameter for the request</td>
</tr>
<tr>
<td>...</td>
<td>Arguments to be passed to the query list for GET</td>
</tr>
</tbody>
</table>

Value

List of entries from SCOPUS

See Also

get_author_info

Examples

```bash
## Not run:
author_search(au_id = "Smith", searcher = "affil(princeton) and authlast")
berk = author_search(au_id = "berkeley", searcher = "affil", count =100)

## End(Not run)
```

author_search_by_affil

Search Authors by Affiliation on SCOPUS

Description

Searches SCOPUS to get information about authors with a certain affiliation

Usage

author_search_by_affil(affil_id, searcher = "AF-ID", ...)

Arguments

<table>
<thead>
<tr>
<th>affil_id</th>
<th>Affiliation ID number</th>
</tr>
</thead>
<tbody>
<tr>
<td>searcher</td>
<td>Identifier for Affiliation ID. Do not change unless you know exactly what the API calls for.</td>
</tr>
<tr>
<td>...</td>
<td>Arguments to be passed to GET</td>
</tr>
</tbody>
</table>

Value

List of entries from SCOPUS
**SCOPUS Citation Retrieval**

**Description**

SCOPUS Citation Retrieval

**Usage**

```r
citation_retrieval(scopus_id = NULL, pii = NULL, doi = NULL, pubmed_id = NULL, date_range = NULL, ...)
parse_citation_retrieval(result)
```

**Arguments**

- **scopus_id**: Scopus Identifier
- **pii**: Scopus Identifier
- **doi**: Scopus Identifier
- **pubmed_id**: Scopus Identifier
- **date_range**: Date range to specify, must be length 2
- **...**: Arguments to be passed to `generic_elsevier_api`
- **result**: Result from `citation_retrieval`, which has an element of content

**Value**

List of elements, similar to `generic_elsevier_api`

**See Also**

- `generic_elsevier_api`

**Examples**

```r
api_key = Sys.getenv("Elsevier_API_Interactive")
set_api_key(api_key)
if (!is.null(api_key) & nchar(api_key) > 0){
  result = citation_retrieval(pii = c("S0140673616324102", "S0014579301033130"), verbose = FALSE)
  if (http::status_code(result$get_statement) < 400) {
    res = parse_citation_retrieval(result)
  }
}
collapse_affil  
*Collapse Affiliations to one row*

**Description**
Take an individual SCOPUS entry and transform it to a data frame of affiliations

**Usage**
collapse_affil(affils, collapse = ";")

**Arguments**
- **affils**: Data frame of affiliations to be collapsed usually from `author_search`
- **collapse**: What should values be collapsed using as a separator

**Value**
A data.frame of affiliation information

---

complete_multi_author_info  
*Get Complete Author Information and ID from Scopus*

**Description**
Get Complete Author Information and ID from Scopus

**Usage**
complete_multi_author_info(au_id = NULL, api_key = NULL, verbose = TRUE, ...)

**Arguments**
- **au_id**: vector of Author IDs
- **api_key**: Elsevier API key
- **verbose**: Print messages from specification
- **...**: options to pass to `generic_elsevier_api`

**Value**
List of information
elsevier_authenticate  

Authenticate API Key and get Token

Description

Authenticate API Key and get Token

Usage

```r
elsevier_authenticate(api_key = NULL, api_key_error = TRUE, 
choice = NULL, verbose = TRUE, ...)
```

Arguments

- **api_key**: Elsevier API key
- **api_key_error**: Should there be an error if no API key?
- **choice**: Choice of which registered See https://dev.elsevier.com/tecdoc_api_authentication.html
- **verbose**: Print messages from specification
- ... Additional arguments to send to GET.

Value

List of content, the GET request, and the token

Examples

```r
if (have_api_key()) {
  auth = elsevier_authenticate()
}
```

embase_retrieval  SCOPUS Embase Retrieval

Description

This function wraps generic_elsevier_api to give a retrieval of an Embase Entry from the Elsevier Embase Retrieval API

Usage

```r
embase_retrieval(id, identifier = c("lui", "pii", "doi", "embase", "pubmed_id", "medline"), http_end = NULL, ...)
```
Arguments

- **id**: Identifier for abstract
- **identifier**: Type of identifier to use
- **http_end**: any additional end to http statement. See *generic_elsevier_api*
- **...**: Arguments to be passed to *generic_elsevier_api*

Value

List of elements, similar to *generic_elsevier_api*

See Also

*generic_elsevier_api*

Examples

```r
api_key = get_api_key(NULL, error = FALSE)
if (!is.null(api_key)){
  x = embase_retrieval("S1053811915002700", identifier = "pii",
                      verbose = FALSE)
}
```

Description

This function wraps *generic_elsevier_api* to give a retrieval of an entitlement from the Elsevier Text Entitlement API

Usage

```r
entitlement_retrieval(id, identifier = c("scopus_id", "eid", "doi",
                                        "pii", "pubmed_id"), http_end = NULL, ...)
```

Arguments

- **id**: Identifier for entitlement
- **identifier**: Type of identifier to use
- **http_end**: any additional end to http statement. See *generic_elsevier_api*
- **...**: Arguments to be passed to *generic_elsevier_api*

Value

List of elements, similar to *generic_elsevier_api*
See Also
generic_elsevier_api

Examples
api_key = get_api_key(NULL, error = FALSE)
if (!is.null(api_key)){
  x = entitlement_retrieval("S105381915002700", identifier = "pii",
                         verbose = FALSE)
}

entries_to_affil_list

List of SCOPUS Entries to List of Affiliations Data Frames

Description
Take a SCOPUS entry and transform it to a data frame of affiliations

Usage
entries_to_affil_list(entries)

Arguments
entries list of entries from SCOPUS, usually from author_search result

Value
A data.frame of affiliation information

entries_to_citation_df

List of SCOPUS Entries to Data Frame of Citations

Description
Take a SCOPUS entry list and transform it to a data frame of citation and article information

Usage
entries_to_citation_df(entries)

Arguments
entries list of entries from SCOPUS, usually from author_search result

Value
A data.frame of citation information
entries_to_df  
Convert Entries into a data.frame

Description
Converting a list of entries into a data.frame of records

Usage
entries_to_df(entries, au_id = NULL, verbose = TRUE)

entries_to_df2(entries, verbose = TRUE)

Arguments
- entries: Entries from the output of a command
- au_id: Author ID number. Overrides any first/last name argument
- verbose: Print diagnostic messages

Value
Data frame of records

entry_to_affil  
SCOPUS Entry to Affiliation

Description
Take an individual SCOPUS entry and transform it to a data frame of affiliations

Usage
entry_to_affil(x, all_affils)

Arguments
- x: individual entry from SCOPUS, usually from author_search
- all_affils: Affiliation data.frame from all_possible_affils

Value
A data.frame of affiliation information
**generic_elsevier_api**  
**Generic Elsevier Search**

**Description**

Runs GET on generic Elsevier Search

**Usage**

```python
import generic_elsevier_api

s = generic_elsevier_api(query = NULL, type = c("search", "article",  
  "entitlement", "recommendation", "object", "fragment", "abstract", 
  "affiliation", "embase", "author", "serial", "nonserial", "subject", 
  "holdings", "citation-count", "citations", "metadata", "ev", 
  "ev_records"), search_type = c("affiliation", "author", "scopus", 
  "scidir", "scidir-object", "sciencedirect"), api_key = NULL, 
  headers = NULL, content_type = c("content", "feedback"), 
  root_http = "https://api.elsevier.com", http_end = NULL, 
  verbose = TRUE, api_key_error = TRUE, ...)  
```

**Arguments**

- **query**  
  Query to run, not overall query, but `queryParam` query

- **type**  
  Type of search. See [https://dev.elsevier.com/api_docs.html](https://dev.elsevier.com/api_docs.html)

- **search_type**  
  Type of search if `type = "search"`. See [https://dev.elsevier.com/api_docs.html](https://dev.elsevier.com/api_docs.html)

- **api_key**  
  Elsevier API key

- **headers**  
  Headers passed to `add_headers`, passed to `GET`

- **content_type**  
  Is the data content or feedback?

- **root_http**  
  address to use for query

- **http_end**  
  string to add to end of http specification (done using `paste0`)

- **verbose**  
  Print messages from specification

- **api_key_error**  
  Should there be an error if no API key?

...  
Options passed to `queryParam` for `GET`

**Value**

List of elements, content and the GET request

**Examples**

```r
## Not run:
query_string = "affil(hopkins)"
# Use affiliation query
s = generic_elsevier_api(query = query_string,  
  type = "search", search_type = "affiliation",
```
get_affiliation_info

```r
# Use author query
s = generic_elsevier_api(query = query_string,
type = "search", search_type = "author",
api_key = api_key,
verbose = FALSE)

# Query abstract by pii
s = generic_elsevier_api(query = "",
type = "abstract", http_end = "pii/S1053811915002700",
api_key = api_key,
verbose = FALSE)
```

## End (Not run)

---

**gen_entries_to_df**  
*Generally Convert Entries into a list of data.frames*

**Description**

Generally Convert Entries into a list of data.frames

**Usage**

```r
gen_entries_to_df(entries, scrub = FALSE)
```

**Arguments**

- `entries`  
  Entries from the output of a command

- `scrub`  
  Should ‘scrub_identifier’ be run on the identifier?

**Value**

List of data.frames from entries

---

**get_affiliation_info**  
*Get Affiliation Information and ID from Scopus*

**Description**

Uses SCOPUS affiliation search to identify affiliation identification information

**Usage**

```r
get_affiliation_info(affil_id = NULL, affil_name = NULL,
api_key = NULL, verbose = FALSE)
```
get_api_key

Arguments

- affil_id: ID of affiliation
- affil_name: name of affiliation
- api_key: Elsevier API key
- verbose: Print messages from specification

Value

A data.frame of affiliation information

Description

Determines if option(elsevier_api_key) or option(elsevier_api_key_filename) is set. If not, it stops and returns an error. If so, returns the value.

Usage

get_api_key(api_key = NULL, error = TRUE)

have_api_key(api_key = NULL)

Arguments

- api_key: Elsevier API key
- error: Should the function error if api_key = NULL?

Value

API key

Note

You can either set the API key using option(elsevier_api_key) or have it accessible by api_key = Sys.getenv('Elsevier_API_key')

Examples

res = get_api_key(error = FALSE)
get_author_info

Get Relevant Author Information and ID from Scopus in Data Frame

Description

Uses SCOPUS author search to identify author identification information in a workable format.

Usage

get_author_info(...)

Arguments

...  Arguments passed to get_complete_author_info

Value

Data.frame of information

See Also

get_complete_author_info

Examples

## Not run:
get_author_info(au_id = "40462056100")

## End(Not run)

get_complete_author_info

Get Complete Author Information and ID from Scopus

Description

Uses SCOPUS author search to identify author identification information.

Usage

get_complete_author_info(last_name = NULL, first_name = NULL,
                        affil_id = NULL, affil_name = NULL, api_key = NULL,
                        http = "https://api.elsevier.com/content/search/author",
                        query = NULL, verbose = TRUE, au_id = NULL, ...)

## Not run:
get_complete_author_info(au_id = "40462056100")

## End(Not run)
get_links

Arguments

- `last_name`: last name of author
- `first_name`: first name of author
- `affil_id`: ID of affiliation (optional)
- `affil_name`: name of affiliation
- `api_key`: Elsevier API key
- `http`: Author API http
- `query`: Additional query info, added using +AND+ to original query
- `verbose`: Print messages from specification
- `au_id`: Author ID number, will override first/last combination if specified
- `...`: options to pass to GET

Value

List of information

Examples

```bash
if (have_api_key()) {
  res = get_complete_author_info(
    last_name = "Muschelli",
    first_name = "John",
    verbose = FALSE)
}
```

---

get_links: Get Links for next/first/last query

Description

Get Links for next/first/last query

Usage

```bash
get_links(result)
get_url(result, type = c("self", "first", "prev", "next", "last"))
next_url(result)
last_url(result)
prev_url(result)
sel_url(result)
```
first_url(result)

get_link_type(result, ..., type = c("self", "first", "prev", "next", "last"))

get_first(result, ...)

get_last(result, ...)

get_prev(result, ...)

get_next(result, ...)

get_self(result, ...)

**Arguments**

- `result` Object (list) with an element named `content`, usually from `generic_elsevier_api`
- `type` The type of link requested
- `...` Options passed to `GET`

**Value**

A `data.frame` or a vector of characters

**Examples**

```r
## Not run:
result <- generic_elsevier_api(
quaray = "ISSN(0004-3702) AND YEAR(2001)",
search_type = "scopus")
next_result = get_next(result)

## End(Not run)
```

---

**Add Institution or Authorization Token**

**Description**

Add Institution or Authorization Token
metadata_retrieval

Usage

inst_token_header(token)

auth_token_header(token)

Arguments

token Elsevier API token, usually from elsevier_authenticate

Value

An object of class request

metadata_retrieval SCOPUS Citation Retrieval

Description

SCOPUS Citation Retrieval

Usage

metadata_retrieval(query = NULL, view = c("STANDARD", "COMPLETE"), ...)

Arguments

query Query to run, not overall query, but 'queryParam' query

view Type of view to have. See https://dev.elsevier.com/guides/ArticleMetadataViews.htm

... Arguments to be passed to generic_elsevier_api

Value

List of elements, similar to generic_elsevier_api

See Also

generic_elsevier_api, https://dev.elsevier.com/documentation/ArticleMetadataAPI.wadl

Examples

api_key = get_api_key(NULL, error = FALSE)
if (!is.null(api_key)){
    x = metadata_retrieval(query = "heart attack",
                            verbose = FALSE)
}
multi_author_info  Get Relevant Authors Information from IDs from Scopus

Description
Get Relevant Authors Information from IDs from Scopus

Usage
multi_author_info(
  ...
)

process_complete_multi_author_info(res)

Arguments
... Arguments passed to get_complete_author_info
res result from complete_multi_author_info

Value
Data.frame of information

See Also
get_complete_author_info

Examples
## Not run:
multi_author_info(au_id = c("22968535800", "40462056100"))

## End(Not run)

nonnull  Remove NULL

Description
Removes NULL values from a vector from a list

Usage
nonnull(x, replace = NA)
Arguments

- x: Vector from a list
- replace: Value to replace NULL with

Value

Vector

---

**object_retrieval**  
*ScienceDirect Object Retrieval*

**Description**

This function wraps `generic_elsevier_api` to give a retrieval of an object from the Elsevier Object Retrieval API.

**Usage**

```r
object_retrieval(id, identifier = c("scopus_id", "eid", "doi", "pii", "pubmed_id"), ref = NULL, http_end = NULL, ...)
```

```r
process_object_retrieval(res)
```

```r
download_object(url, api_key = NULL, api_key_error = TRUE, verbose = TRUE, ...)
```

```r
download_objects(url, ...)
```

**Arguments**

- id: Identifier for object
- identifier: Type of identifier to use
- ref: document reference
- http_end: any additional end to http statement. See `generic_elsevier_api`
- ...: Arguments to be passed to `generic_elsevier_api` or GET
- res: result from `object_retrieval`
- url: url to download from `object_retrieval`
- api_key: Elsevier API key
- api_key_error: Should there be an error if no API key?
- verbose: Print messages from specification

**Value**

List of elements, similar to `generic_elsevier_api`
See Also

generic_elsevier_api

Examples

api_key = get_api_key(NULL, error = FALSE)
if (!is.null(api_key)) {
  x = object_retrieval("S10538111915002700", identifier = "pii",
                      verbose = FALSE)
  df = process_object_retrieval(x)
  df = df[grepl("image/jpeg", df$mime_type), drop = FALSE]
  df = df[df$type %in% "IMAGE-HIGH-RES", drop = FALSE]
  res = download_object(df$url[1])
  if (interactive()) {
    browseURL(res$outfile)
  } else {
    img = res$content
    dims = dim(img)[1:2]
    ndim = max(dims)
    graphics::plot(c(0, ncol(img)), c(0, nrow(img)), type='n')
    graphics::rasterImage(img, 1, 1, ncol(img), nrow(img))
  }
}
**process_author_name**  

*Process Author Name*

**Description**

Process author ID and names for generic use

**Usage**

```python
process_author_name(au_id = NULL, last_name = NULL,  
first_name = NULL, affil_id = NULL, api_key = NULL,  
verbose = TRUE)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>au_id</td>
<td>Author ID number. Overrides any first/last name argument</td>
</tr>
<tr>
<td>last_name</td>
<td>last name of author</td>
</tr>
<tr>
<td>first_name</td>
<td>first name of author</td>
</tr>
<tr>
<td>affil_id</td>
<td>ID of affiliation (optional)</td>
</tr>
<tr>
<td>api_key</td>
<td>Elsevier API key</td>
</tr>
<tr>
<td>verbose</td>
<td>Print diagnostic messages</td>
</tr>
</tbody>
</table>

**Value**

List of affiliation name and affiliation ID
**Value**

List of first/last name and author ID

**Note**

This function is really to avoid duplication

---

**read_cto**

*Read Citation Overview (CTO) File*

**Description**

Read Citation Overview (CTO) File

**Usage**

```
read_cto(file)
read_cto_long(file)
```

**Arguments**

- **file**: CSV of CTO export from Scopus

**Value**

A list of the data, year columns, and header information

**Examples**

```
file = system.file("extdata", "CTOExport.csv", package = "rscopus")
citations_over_time = read_cto(file)
citations_over_time = citations_over_time$data
```

---

**recommendation_retrieval**

*ScienceDirect Article Recommendation Retrieval*

**Description**

This function wraps `generic_elsevier_api` to give a retrieval of a recommendation from the Elsevier Article Recommendation API

**Usage**

```
recommendation_retrieval(id, identifier = c("eid", "pii"),
http_end = NULL, ...)
```
replace_non_ascii

Arguments

- id: Identifier for recommendation
- identifier: Type of identifier to use
- http_end: any additional end to http statement. See `generic_elsevier_api`
- ...: Arguments to be passed to `generic_elsevier_api`

Value

List of elements, similar to `generic_elsevier_api`

See Also

`generic_elsevier_api`

Examples

```r
api_key = get_api_key(NULL, error = FALSE)
if (!is.null(api_key)){
  x = recommendation_retrieval("S1053811915002700", identifier = "pii",
    verbose = FALSE)
}
```

---

replace_non_ascii **Replace non-ASCII characters**

Description

Replaces non-ASCII characters from a last or first name

Usage

`replace_non_ascii(string)`

Arguments

- string: Character vector of values to be replaced

Value

Character vector

See Also

Taken from [https://stackoverflow.com/questions/20495598/replace-accented-characters-in-r-with-non-ac](https://stackoverflow.com/questions/20495598/replace-accented-characters-in-r-with-non-ac)
Description

This function wraps generic_elsevier_api to give a scopus search from the Elsevier Scopus Search API.

Searches SCOPUS to get information about documents on an author.

Usage

```
scopus_search(query, api_key = NULL, count = 200,
              view = c("STANDARD", "COMPLETE"), start = 0, verbose = TRUE,
              max_count = 20000,
              http = "https://api.elsevier.com/content/search/scopus", ...)
```

```
sciencedirect_search(count = 100, ...)
```

```
escidir_search(count = 100, ...)
```

Arguments

- **query**
  - Query string to search on SCOPUS
- **api_key**
  - API Key for Elsevier
- **count**
  - number of records to retrieve (below 200 for STANDARD, below 25 for COMPLETE views, see https://dev.elsevier.com/api_key_settings.html).
- **view**
  - type of view to give, see https://api.elsevier.com/documentation/ScopusSearchAPI.wadl
- **start**
  - where should the records start gathering
- **verbose**
  - Print diagnostic messages
- **max_count**
  - Maximum count of records to be returned.
- **http**
  - Address for scopus API
- **...**
  - Arguments to be passed to the query list for GET

Value

List of entries from SCOPUS

Examples

```
if (have_api_key()) {
  res = scopus_search(query = "all(gene)", max_count = 20,
                      count = 10)
  df = gen_entries_to_df(res$entries)
  head(df$df)
}
set_api_key

sci_res = sciencedirect_search(query = "heart+attack AND text(liver)",
max_count = 30, count = 25)
sci_df = gen_entries_to_df(sci_res$entries)

nt = sciencedirect_search(query = "title(neurotoxin)", max_count = 20,
count = 10)
nt_df = gen_entries_to_df(nt$entries)
nt_df = nt_df$df

Description
Set API Key for Elsevier

Sets Elsevier API key using if option(elsevier_api_key)

Usage
set_api_key(api_key)

Arguments
api_key Elsevier API key

subject_areas

Description
Subject Areas

Usage
subject_areas()

subject_area_codes()

Value
Character vector of subject areas

Note
See https://service.elsevier.com/app/answers/detail/a_id/15181/supporthub/scopus/related/1/
Index

abstract_retrieval, 3
add_headers, 19
affil_data (affil_df), 5
affil_df, 5
affil_list_to_df, 6
affiliation_retrieval, 4
all_possible_affils, 7, 18
article_retrieval, 7
auth_token_header (inst_token_header), 24
author_data (author_df), 8
author_df, 8
author_df_orig (author_df), 8
author_list (author_df), 8
author_retrieval (author_retrieval_id), 10
author_retrieval_id, 10
author_search, 5–7, 9, 11, 14, 17, 18
author_search_by_affil, 12
citation_retrieval, 13, 13
collapse_affil, 14
copyright_multi_author_info, 14, 26
download_object (object_retrieval), 27
download_objects (object_retrieval), 27
elsevier_authenticate, 15, 25
embase_retrieval, 15
entitlement_retrieval, 16
entries_to_affil_list, 6, 17
entries_to_citation_df, 17
entries_to_df, 18
entries_to_df2 (entries_to_df), 18
entry_to_affil, 18
first_url (get_links), 23
gen_entries_to_df, 9, 20
generic_elsevier_api, 3, 4, 7, 8, 10, 11, 13–17, 19, 25, 27, 28, 30–32
GET, 12, 15, 19, 23, 24, 27, 32
get_author_info, 20
general_key, 21
get_author_info, 5, 7, 9, 12, 13, 22
get_complete_author_info, 22, 22, 26
get_first (get_links), 23
get_last (get_links), 23
general_type (get_links), 23
general_links, 23
general_next (get_links), 23
general_prev (get_links), 23
general_self (get_links), 23
general_url (get_links), 23
have_api_key (get_api_key), 21
inst_token_header, 24
last_url (get_links), 23
metadata_retrieval, 25
multi_author_info, 26
multi_author_retrieval
  (author_retrieval_id), 10
next_url (get_links), 23
nonull, 26
object_retrieval, 27, 27
parse_citation_retrieval
  (citation_retrieval), 13
prev_url (get_links), 23
print Scopus_api_key, 28
process_author_info, 29
process_author_name, 29
process_complete_multi_author_info
  (multi_author_info), 26
INDEX

process_object_retrieval
  (object_retrieval), 27

read_cto, 30
read_cto_long(read_cto), 30
recommendation_retrieval, 30
replace_non_ascii, 31

scidir_search(scopus_search), 32
sciedirect_search(scopus_search), 32
scopus_search, 32
self_url(get_links), 23
set_api_key, 33
subject_area_codes(subject_areas), 33
subject_areas, 33