

# Package ‘rscopus’

October 14, 2022

**Type** Package

**Title** Scopus Database 'API' Interface

**Version** 0.6.6

**Date** 2019-09-17

**Maintainer** John Muschelli <muschellij2@gmail.com>

**Description** Uses Elsevier 'Scopus' API  
<[https://dev.elsevier.com/sc\\_apis.html](https://dev.elsevier.com/sc_apis.html)> to download  
information about authors and their citations.

**License** GPL-2

**Depends** R (>= 3.0.0)

**Imports** httr, jsonlite, utils, stats, plyr, tidyr, dplyr, tools, glue

**LazyData** TRUE

**Suggests** xml2, rvest, graphics, testthat, jpeg, knitr, rmarkdown,  
purrr

**URL** [https://dev.elsevier.com/sc\\_apis.html](https://dev.elsevier.com/sc_apis.html),  
<https://github.com/muschellij2/rscopus>

**BugReports** <https://github.com/muschellij2/rscopus/issues>

**RoxygenNote** 6.1.1

**Encoding** UTF-8

**VignetteBuilder** knitr

**NeedsCompilation** no

**Author** John Muschelli [aut, cre]

**Repository** CRAN

**Date/Publication** 2019-09-17 20:20:02 UTC

**R topics documented:**

abstract_retrieval	3
affiliation_retrieval	4
affil_df	5
affil_list_to_df	6
affil_search	6
all_possible_affils	7
article_retrieval	7
author_df	8
author_retrieval_id	10
author_search	11
author_search_by_affil	12
bibtex_core_data	13
citation_retrieval	14
collapse_affil	15
complete_multi_author_info	15
elsevier_authenticate	16
embase_retrieval	16
entitlement_retrieval	17
entries_to_affil_list	18
entries_to_citation_df	18
entries_to_df	19
entry_to_affil	19
generic_elsevier_api	20
gen_entries_to_df	21
get_affiliation_info	21
get_api_key	22
get_author_info	23
get_complete_author_info	23
get_links	24
inst_token_header	25
metadata_retrieval	26
multi_author_info	27
nonull	27
object_retrieval	28
plumx_metrics	29
print.scopus_api_key	30
print.token	31
process_affiliation_name	31
process_author_name	32
read_cto	33
recommendation_retrieval	33
replace_non_ascii	34
scopus_search	35
set_api_key	36
subject_areas	36

---

abstract_retrieval	<i>SCOPUS Abstract Retrieval</i>
--------------------	----------------------------------

---

### 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 <a href="#">generic_elsevier_api</a>
...	Arguments to be passed to <a href="#">generic_elsevier_api</a>

### 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("S1053811915002700", identifier = "pii",  
    verbose = FALSE)  
  x = abstract_retrieval("S1053811915002700", identifier = "pii",  
    view = "FULL",  
    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

<code>id</code>	Identifier for affiliation
<code>identifier</code>	Type of identifier to use
<code>http_end</code>	any additional end to http statement. See <a href="#">generic_elsevier_api</a>
<code>...</code>	Arguments to be passed to <a href="#">generic_elsevier_api</a>

### 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**

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

**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	<i>List of SCOPUS Entries to List of Affiliations Data Frames</i>
------------------	---

---

**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	<i>Search Author Content on SCOPUS</i>
--------------	--

---

**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 <a href="https://dev.elsevier.com/api_docs.html">https://dev.elsevier.com/api_docs.html</a>
count	number of records to retrieve (below 200, see <a href="https://dev.elsevier.com/api_key_settings.html">https://dev.elsevier.com/api_key_settings.html</a> )
...	Arguments to be passed to <a href="#">author_search</a>

**Value**

List of entries from SCOPUS

**See Also**

[get\\_author\\_info](#)

---

*all\_possible\_affils*     *Find all affiliations*

---

**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, ...)
```

**Arguments**

id	Identifier for article
view	Which view to see. See <a href="https://dev.elsevier.com/guides/ArticleRetrievalViews.htm">https://dev.elsevier.com/guides/ArticleRetrievalViews.htm</a>
identifier	Type of identifier to use
http_end	any additional end to http statement. See <a href="#">generic_elsevier_api</a>
...	Arguments to be passed to <a href="#">generic_elsevier_api</a>

**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 = 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**

```
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,
  headers = NULL, ...)

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



```
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, headers = NULL, ...)

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 <a href="https://api.elsevier.com/documentation/ScopusSearchAPI.wadl">https://api.elsevier.com/documentation/ScopusSearchAPI.wadl</a>
count	number of records to retrieve (below 25, see <a href="https://dev.elsevier.com/api_key_settings.html">https://dev.elsevier.com/api_key_settings.html</a> )
general	Should <a href="#">gen_entries_to_df</a> instead of the way before version 0.5.10.9001
scrub	Should 'scrub_identifier' be run on the identifier?
headers	Headers passed to <a href="#">add_headers</a> , passed to GET
...	Arguments to be passed to <a href="#">author_search</a>

### 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, headers = NULL, ...)
```

### Arguments

id	Identifier for author
identifier	Type of identifier to use
http_end	any additional end to http statement. See <a href="#">generic_elsevier_api</a>
...	Arguments to be passed to <a href="#">generic_elsevier_api</a>
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 <a href="https://api.elsevier.com/documentation/AuthorRetrievalAPI.wadl">https://api.elsevier.com/documentation/AuthorRetrievalAPI.wadl</a>
self_cite	Should self-citations be included?
verbose	Print diagnostic messages
api_key	Elsevier API key
headers	Headers passed to <a href="#">add_headers</a> , passed to <a href="#">GET</a>

### Value

List of elements, similar to [generic\\_elsevier\\_api](#)

### Note

See <https://api.elsevier.com/documentation/AuthorRetrievalAPI.wadl> for documentation

**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, headers = NULL, wait_time = 0, ...)
```

**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 <a href="https://dev.elsevier.com/api_key_settings.html">https://dev.elsevier.com/api_key_settings.html</a> )
start	where should the records start gathering
verbose	Print diagnostic messages
facets	Facets sent in query. See <a href="https://dev.elsevier.com/api_docs.html">https://dev.elsevier.com/api_docs.html</a>

searcher	Identifier for author ID. Do not change unless you know exactly what the API calls for.
max_count	Maximum count of records to be returned.
view	type of view to give, see <a href="https://api.elsevier.com/documentation/AuthorSearchAPI.wadl">https://api.elsevier.com/documentation/AuthorSearchAPI.wadl</a>
add_query	Things to add to the query parameter for the request
headers	additional headers to be added to <a href="#">add_headers</a>
wait_time	The time in seconds to wait across consecutive requests of a single search (when records > 25)
...	Arguments to be passed to the query list for <a href="#">GET</a>

**Value**

List of entries from SCOPUS

**See Also**

[get\\_author\\_info](#)

**Examples**

```
## 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**

affil_id	Affiliation ID number
searcher	Identifier for Affiliation ID. Do not change unless you know exactly what the API calls for.
...	Arguments to be passed to <a href="#">GET</a>

**Value**

List of entries from SCOPUS

**See Also**

[get\\_author\\_info](#)

---

bibtex_core_data	Makes a bibtex entry from an output of <a href="#">abstract_retrieval</a> or <a href="#">article_retrieval</a>
------------------	--

---

**Description**

Makes a bibtex entry from an output of [abstract\\_retrieval](#) or [article\\_retrieval](#)

**Usage**

```
bibtex_core_data(x)
```

**Arguments**

x                    output of [abstract\\_retrieval](#) or [article\\_retrieval](#), with both `get_statement` and `content`

**Value**

A character vector of bibtex values

**Note**

Adapted from [https://github.com/pybliometrics-dev/pybliometrics/blob/master/pybliometrics/scopus/abstract\\_retrieval.py](https://github.com/pybliometrics-dev/pybliometrics/blob/master/pybliometrics/scopus/abstract_retrieval.py)

**Examples**

```
api_key = get_api_key(NULL, error = FALSE)
if (!is.null(api_key)){
  x = abstract_retrieval("S1053811915002700", identifier = "pii",
    verbose = FALSE)
  res = bibtex_core_data(x)
  cat(res)
  x = abstract_retrieval("S1053811915002700", identifier = "pii",
    verbose = FALSE)
  res2 = bibtex_core_data(x)
  cat(res2)
}
```

---

citation\_retrieval      *SCOPUS Citation Retrieval*

---

**Description**

SCOPUS Citation Retrieval

**Usage**

```
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 <a href="#">generic_elsevier_api</a>
result	result from <a href="#">citation_retrieval</a> , which has an element of content

**Value**

List of elements, similar to [generic\\_elsevier\\_api](#)

**See Also**

[generic\\_elsevier\\_api](#)

**Examples**

```
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 (httr::status_code(result$get_statement) < 400) {
    res = parse_citation_retrieval(result)
  }
}
set_api_key(NULL)
```

---

collapse_affil	<i>Collapse Affiliations to one row</i>
----------------	---

---

**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 <a href="#">author_search</a>
collapse	What should values be collapsed using as a separator

**Value**

A data.frame of affiliation information

---

complete_multi_author_info	<i>Get Complete Author Information and ID from Scopus</i>
----------------------------	---

---

**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 <a href="#">generic_elsevier_api</a>

**Value**

List of information

---

elsevier\_authenticate *Authenticate API Key and get Token*

---

### Description

Authenticate API Key and get Token

### Usage

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

### Arguments

api_key	Elsevier API key
api_key_error	Should there be an error if no API key?
choice	Choice of which registered See <a href="https://dev.elsevier.com/tecdoc_api_authentication.html">https://dev.elsevier.com/tecdoc_api_authentication.html</a>
verbose	Print messages from specification
headers	Headers passed to <a href="#">add_headers</a> , passed to <a href="#">GET</a>
...	Additional arguments to send to <a href="#">GET</a> .

### Value

List of content, the GET request, and the token

### Examples

```
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

```
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 <a href="#">generic_elsevier_api</a>
...	Arguments to be passed to <a href="#">generic_elsevier_api</a>

**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 = embase_retrieval("S1053811915002700", identifier = "pii",
    verbose = FALSE)
}
```

---

entitlement\_retrieval *ScienceDirect Text Entitlement Retrieval*

---

**Description**

This function wraps [generic\\_elsevier\\_api](#) to give a retrieval of an entitlement from the Elsevier Text Entitlement API

**Usage**

```
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 <a href="#">generic_elsevier_api</a>
...	Arguments to be passed to <a href="#">generic_elsevier_api</a>

**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("S1053811915002700", 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	<i>Convert Entries into a data.frame</i>
---------------	--

---

**Description**

Converts 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	<i>SCOPUS Entry to Affiliation</i>
----------------	------------------------------------

---

**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 <a href="#">author_search</a>
all_affils	Affiliation data.frame from <a href="#">all_possible_affils</a>

**Value**

A data.frame of affiliation information

---

generic\_elsevier\_api    *Generic Elsevier Search*

---

### Description

Runs GET on generic Elsevier Search

### Usage

```
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", "analytics"), search_type = c("affiliation", "author",
  "scopus", "scidir", "scidir-object", "sciencedirect", "plumx"),
  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 <a href="https://dev.elsevier.com/api_docs.html">https://dev.elsevier.com/api_docs.html</a>
search_type	Type of search if type = "search". See <a href="https://dev.elsevier.com/api_docs.html">https://dev.elsevier.com/api_docs.html</a>
api_key	Elsevier API key
headers	Headers passed to <a href="#">add_headers</a> , passed to <a href="#">GET</a>
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 <a href="#">GET</a>

### Value

List of elements, content and the GET request

### Examples

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

```

        api_key = api_key,
        verbose = FALSE)

# 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

```
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

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

**Arguments**

affil_id	ID of affiliation
affil_name	name of affiliation
api_key	Elsevier API key
verbose	Print messages from specification
headers	Headers passed to <a href="#">add_headers</a> , passed to <a href="#">GET</a>

**Value**

A data.frame of affiliation information

---

get_api_key	<i>Find API Key for Elsevier</i>
-------------	----------------------------------

---

**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 <code>api_key = NULL</code> ?

**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')`.

**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, count = 200, start = 0, verbose = TRUE,  
  au_id = NULL, headers = NULL, ...)
```

**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
count	maximum number of records to retrieve
start	index to start on. Only necessary if a large number of records retrieved
verbose	Print messages from specification
au_id	Author ID number, will override first/last combination if specified
headers	Headers passed to <a href="#">add_headers</a> , passed to <a href="#">GET</a>
...	options to pass to <a href="#">GET</a>

**Value**

List of information

**Examples**

```
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**

```
get_links(result)

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

next_url(result)

last_url(result)
```



```

prev_url(result)
self_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 <a href="#">GET</a>

### Value

A data.frame or a vector of characters

### Examples

```

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

## End(Not run)

```

---

inst_token_header	<i>Add Institution or Authorization Token</i>
-------------------	---

---

### Description

Add Institution or Authorization Token

**Usage**

```
inst_token_header(token)
```

```
auth_token_header(token)
```

**Arguments**

token                    Elsevier API token, usually from [elsevier\\_authenticate](#)

**Value**

An object of class token, but really a character

---

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	<i>Get Relevant Authors Information from IDs from Scopus</i>
-------------------	--

---

**Description**

Get Relevant Authors Information from IDs from Scopus

**Usage**

```
multi_author_info(...)  
  
process_complete_multi_author_info(res)
```

**Arguments**

...	Arguments passed to <a href="#">get_complete_author_info</a>
res	result from <a href="#">complete_multi_author_info</a>

**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)
```

---

nonull	<i>Remove NULL</i>
--------	--------------------

---

**Description**

Removes NULL values from a vector from a list

**Usage**

```
nonull(x, replace = NA)
```

**Arguments**

x	Vector from a list
replace	Value to replace NULL with

**Value**

Vector

---

object_retrieval	<i>ScienceDirect Object Retrieval</i>
------------------	---------------------------------------

---

**Description**

This function wraps [generic\\_elsevier\\_api](#) to give a retrieval of an object from the Elsevier Object Retrieval API

**Usage**

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

```
process_object_retrieval(res)
```

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

```
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 <a href="#">generic_elsevier_api</a>
...	Arguments to be passed to <a href="#">generic_elsevier_api</a> or <a href="#">GET</a>
res	result from <a href="#">object_retrieval</a>
url	url to download from <a href="#">object_retrieval</a>
api_key	Elsevier API key
api_key_error	Should there be an error if no API key?
verbose	Print messages from specification
headers	Headers passed to <a href="#">add_headers</a> , passed to <a href="#">GET</a>

**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("S1053811915002700", 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]
    mdim = max(dims)
    graphics::plot(c(0, ncol(img)), c(0, nrow(img)), type='n')
    graphics::rasterImage(img, 1, 1, ncol(img), nrow(img))
  }
}
```

---

plumx_metrics	<i>Retrieve PlumX metrics for Scopus documents and other related artifacts</i>
---------------	--

---

**Description**

Retrieve PlumX metrics for Scopus documents and other related artifacts

**Usage**

```
plumx_metrics(value, type = plumx_types(), ...)
```

```
plumx_types()
```

**Arguments**

value	The value of the identifier to search for.
type	The type of identifier to search for.
...	Additional arguments to pass to <a href="#">generic_elsevier_api</a> , other than <code>http_end</code> , <code>type</code> , <code>search_type</code> , and <code>content_type</code>

**Value**

List of elements, content and the GET request

**Note**

See <https://dev.elsevier.com/documentation/PlumXMetricsAPI.wadl> for more information

**Examples**

```
if (have_api_key()) {  
  type = "doi"  
  value = "10.1016/j.nicl.2018.10.013"  
  res = plumx_metrics(value, type)  
}
```

---

print.scopus\_api\_key *Print method for Scopus API key*

---

**Description**

Print method for Scopus API key

**Usage**

```
## S3 method for class 'scopus_api_key'  
print(x, reveal = FALSE, ...)
```

**Arguments**

x	an object used to select a method.
reveal	Should the API key be revealed
...	further arguments passed to or from other methods

**Examples**

```
x = "asdf"  
class(x) = "scopus_api_key"  
print(x)  
print(x, reveal = TRUE)
```

---

print.token	<i>Print method for token</i>
-------------	-------------------------------

---

**Description**

Print method for token

**Usage**

```
## S3 method for class 'token'  
print(x, reveal = FALSE, ...)  
  
reveal(x, ...)
```

**Arguments**

x	an object used to select a method.
reveal	Should the token be revealed
...	further arguments passed to or from other methods

**Examples**

```
x = "asdf"  
class(x) = "token"  
print(x)  
print(x, reveal = TRUE)
```

---

process_affiliation_name	<i>Process Affiliation Name</i>
--------------------------	---------------------------------

---

**Description**

Process Affiliation ID and names for generic use

**Usage**

```
process_affiliation_name(affil_id = NULL, affil_name = NULL,  
  api_key = NULL, verbose = TRUE)
```

**Arguments**

affil_id	Affiliation ID number.
affil_name	affiliation name
api_key	Elsevier API key
verbose	Print diagnostic messages

**Value**

List of affiliation name and affiliation ID

---

<code>process_author_name</code>	<i>Process Author Name</i>
----------------------------------	----------------------------

---

**Description**

Process author ID and names for generic use

**Usage**

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

**Arguments**

<code>au_id</code>	Author ID number. Overrides any first/last name argument
<code>last_name</code>	last name of author
<code>first_name</code>	first name of author
<code>affil_id</code>	ID of affiliation (optional)
<code>api_key</code>	Elsevier API key
<code>verbose</code>	Print diagnostic messages
<code>headers</code>	Headers passed to <a href="#">add_headers</a> , passed to <a href="#">GET</a>

**Value**

List of first/last name and author ID

**Note**

This function is really to avoid duplication



---

read_cto	<i>Read Citation Overview (CTO) File</i>
----------	--

---

**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	<i>ScienceDirect Article Recommendation Retrieval</i>
--------------------------	---

---

**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, ...)
```

**Arguments**

id	Identifier for recommendation
identifier	Type of identifier to use
http_end	any additional end to http statement. See <a href="#">generic_elsevier_api</a>
...	Arguments to be passed to <a href="#">generic_elsevier_api</a>

**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 = recommendation_retrieval("S1053811915002700", identifier = "pii",
  verbose = FALSE)
}
```

---

replace_non_ascii	<i>Replace non-ASCII characters</i>
-------------------	-------------------------------------

---

**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>

---

scopus_search	<i>SCOPUS Search</i>
---------------	----------------------

---

### 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",
              headers = NULL, wait_time = 0, ...)
```

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

```
scidir_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 <a href="https://dev.elsevier.com/api_key_settings.html">https://dev.elsevier.com/api_key_settings.html</a> ).
view	type of view to give, see <a href="https://api.elsevier.com/documentation/ScopusSearchAPI.wadl">https://api.elsevier.com/documentation/ScopusSearchAPI.wadl</a>
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
headers	additional headers to be added to <a href="#">add_headers</a>
wait_time	The time in seconds to wait across consecutive requests of a single search (when records > 25)
...	Arguments to be passed to the query list for <a href="#">GET</a>

### 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)
  sci_res = sciencedirect_search(query = "heart+attack AND text(liver)",
    max_count = 30, count = 25)
  sci_df = gen_entries_to_df(sci_res$entries)
  Sys.sleep(2)
  nt = sciencedirect_search(query = "title(neurotoxin)", max_count = 20,
    count = 10, wait_time = 1)
  nt_df = gen_entries_to_df(nt$entries)
  nt_df = nt_df$df
}

```

---

set_api_key	<i>Set API Key for Elsevier</i>
-------------	---------------------------------

---

**Description**

Sets Elsevier API key using `if option(elsevier_api_key)`

**Usage**

```
set_api_key(api_key)
```

**Arguments**

api_key	Elsevier API key
---------	------------------

---

subject_areas	<i>Subject Areas</i>
---------------	----------------------

---

**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/](https://service.elsevier.com/app/answers/detail/a_id/15181/supporthub/scopus/related/1/)

# Index

abstract\_retrieval, [3](#), [13](#)  
add\_headers, [9](#), [10](#), [12](#), [16](#), [20](#), [22](#), [24](#), [28](#), [32](#),  
[35](#)  
affil\_data (affil\_df), [5](#)  
affil\_df, [5](#)  
affil\_list\_to\_df, [6](#)  
affil\_search, [6](#)  
affiliation\_retrieval, [4](#)  
all\_possible\_affils, [7](#), [19](#)  
article\_retrieval, [7](#), [13](#)  
auth\_token\_header (inst\_token\_header),  
[25](#)  
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](#), [15](#), [18](#), [19](#)  
author\_search\_by\_affil, [12](#)  
  
bibtex\_core\_data, [13](#)  
  
citation\_retrieval, [14](#), [14](#)  
collapse\_affil, [15](#)  
complete\_multi\_author\_info, [15](#), [27](#)  
  
download\_object (object\_retrieval), [28](#)  
download\_objects (object\_retrieval), [28](#)  
  
elsevier\_authenticate, [16](#), [26](#)  
embase\_retrieval, [16](#)  
entitlement\_retrieval, [17](#)  
entries\_to\_affil\_list, [6](#), [18](#)  
entries\_to\_citation\_df, [18](#)  
entries\_to\_df, [19](#)  
entries\_to\_df2 (entries\_to\_df), [19](#)  
entry\_to\_affil, [19](#)  
  
first\_url (get\_links), [24](#)  
  
gen\_entries\_to\_df, [9](#), [21](#)  
generic\_elsevier\_api, [3](#), [4](#), [7](#), [8](#), [10](#), [11](#),  
[14–18](#), [20](#), [26](#), [28](#), [29](#), [33–35](#)  
GET, [9](#), [10](#), [12](#), [16](#), [20](#), [22](#), [24](#), [25](#), [28](#), [32](#), [35](#)  
get\_affiliation\_info, [21](#)  
get\_api\_key, [22](#)  
get\_author\_info, [5](#), [7](#), [9](#), [12](#), [13](#), [23](#)  
get\_complete\_author\_info, [23](#), [23](#), [27](#)  
get\_first (get\_links), [24](#)  
get\_last (get\_links), [24](#)  
get\_link\_type (get\_links), [24](#)  
get\_links, [24](#)  
get\_next (get\_links), [24](#)  
get\_prev (get\_links), [24](#)  
get\_self (get\_links), [24](#)  
get\_url (get\_links), [24](#)  
  
have\_api\_key (get\_api\_key), [22](#)  
  
inst\_token\_header, [25](#)  
  
last\_url (get\_links), [24](#)  
  
metadata\_retrieval, [26](#)  
multi\_author\_info, [27](#)  
multi\_author\_retrieval  
(author\_retrieval\_id), [10](#)  
  
next\_url (get\_links), [24](#)  
nonnull, [27](#)  
  
object\_retrieval, [28](#), [28](#)  
  
parse\_citation\_retrieval  
(citation\_retrieval), [14](#)  
plumx\_metrics, [29](#)  
plumx\_types (plumx\_metrics), [29](#)  
prev\_url (get\_links), [24](#)  
print.scopus\_api\_key, [30](#)  
print.token, [31](#)  
process\_affiliation\_name, [31](#)

process\_author\_name, [32](#)  
process\_complete\_multi\_author\_info  
    (multi\_author\_info), [27](#)  
process\_object\_retrieval  
    (object\_retrieval), [28](#)

read\_cto, [33](#)  
read\_cto\_long (read\_cto), [33](#)  
recommendation\_retrieval, [33](#)  
replace\_non\_ascii, [34](#)  
reveal (print.token), [31](#)

scidir\_search (scopus\_search), [35](#)  
sciencedirect\_search (scopus\_search), [35](#)  
scopus\_search, [35](#)  
self\_url (get\_links), [24](#)  
set\_api\_key, [36](#)  
subject\_area\_codes (subject\_areas), [36](#)  
subject\_areas, [36](#)