Package ‘rif’

May 16, 2017

Title Client for ‘Neuroscience’ Information Framework ‘APIs’


Version 0.2.0

License MIT + file LICENSE

URL https://github.com/ropensci/rif

BugReports https://github.com/ropensci/rif/issues

VignetteBuilder knitr

Imports jsonlite, crul (>= 0.3.4), data.table, tibble

Suggests roxygen2 (>= 6.0.1), testthat, knitr

RoxygenNote 6.0.1

NeedsCompilation no

Author Scott Chamberlain [aut, cre]

Maintainer Scott Chamberlain <myrmecocystus+r@gmail.com>

Repository CRAN

Date/Publication 2017-05-16 05:34:58 UTC

R topics documented:

rif-package ................................................................. 2
federation_data ........................................................... 3
federation_facet ............................................................ 4
federation_search ......................................................... 6
lexical_chunks ............................................................ 7
lexical_entities ........................................................... 8
lexical_pmchunks ......................................................... 8
lexical_pmentities ....................................................... 9
lexical_pos ............................................................... 10
Description

R Client for Neuroscience Information Framework APIs

API key

You now need an API key to use the NIF API. Steps:

- Go to https://www.neuinfo.org/
- If you don’t have an account yet, register for an account.
- Log in
- Go to https://www.neuinfo.org/account/developer - you’ll need to enter your password again
- Create an API key
- Store your API key ideally as an environment variable in .Renviron or similar. Or, you can save as an R option. (see ?Startup for help on env vars and R options). You can also pass in your API key to each function but that’s not ideal.

Note

The following services are not available in this client:

- annotate - It appears that this is mostly for browser based annotation, so not a good fit for an R package

Author(s)

<myrmecocystus@gmail.com>
Description
Federation data

Usage

federation_data(id, query = NULL, strict = FALSE, subclassQuery = NULL, expandSynonyms = TRUE, expandAcronyms = FALSE, expandAbbrevs = FALSE, expandInferred = TRUE, highlight = FALSE, sortField = NULL, sortAsc = TRUE, offset = 0, count = 20, project = NULL, facet = NULL, filter = NULL, subclassFilter = NULL, includePrimaryData = FALSE, exportType = NULL, orMultiFacets = FALSE, key = NULL, ...)

Arguments

id (character) A dataset id
query Query terms
strict (logical) Only consider more relevant columns. Default: FALSE
subclassQuery (character) An known ID to query with subclass axiom entailment (include all subclasses in the search). This can be combined with the "query" parameter.
expandSynonyms (logical) Should recognized terms in the query have synonyms expanded? Default: TRUE
expandAcronyms (logical) Default: FALSE
expandAbbrevs (logical) Default: FALSE
expandInferred (logical) Should recognized terms in the query have inferred terms expanded? Default: TRUE
highlight (logical) Highlight ...
sortField (character) The field to sort by
sortAsc (logical) Sort ascending?
offset (integer) The result (row) to start on
count (integer) The number of results (rows) to return
project (character) The columns to project (all by default)
facet (character) Facets specified like facetName:facetValue
filter (character) Filters specified like columnName:filterValue. If the column is declared numeric then the filterValue can be a range expression (ie: >24.3).
subclassFilter (character) Filter specified like columnName:parentId where parentId is a known ID. All subclasses of parentId will be included in the filter. This parameter may be combined with "filter".
includePrimaryData

(logical) deprecated Should primary data be included in addition to the view? Please use the exportType option instead.

exportType

(character) The type of data to return: "display" (data suitable for a user interface - includes HTML), "data" (just the raw data used to include the view - may include more data than present in the view but will not include markup) "all" (include both display and data columns)

或多Facets

(logical) True if multiple facet values for the same facet are specified in the query and should be ORed together. Defaults to "AND"

key

(character) API key. required either passed in here or save as an env var or R option. see rif and Startup

... CUrl options passed on to HttpClient

See Also

Other federation: federation_facet, federation_search

Examples

```r
## Not run:
out <- federation_data(id = "nlx_152871-2")
out$query
out$message
out$result$resultCount
out$result$nifid
out$result$debugInfo
out$result$result
out$result$results

federation_data(id = "nlx_152871-2", query = "TS23")

federation_data("nlx_152871-2", verbose = TRUE)

## End(Not run)
```

federation_facet Federation facets

Description

Federation facets

Usage

federation_facet(id, query = NULL, strict = FALSE, subclassQuery = NULL, expandSynonyms = TRUE, expandAcronyms = FALSE, expandAbbrevs = FALSE, expandInferred = TRUE, offset = 0, count = 20, minCount = 2, facet = NULL, filter = NULL, orMultiFacets = FALSE, subclassFilters = NULL, key = NULL, ...)
Arguments

- **id** (character) A dataset id
- **query** (character) A dataset id
- **strict** (logical) Only consider more relevant columns. Default: FALSE
- **subclassQuery** (character) An known ID to query with subclass axiom entailment (include all subclasses in the search). This can be combined with the "query" parameter.
- **expandSynonyms** (logical) Should recognized terms in the query have synonyms expanded? Default: TRUE
- **expandAcronyms** (logical) Default: FALSE
- **expandAbbrevs** (logical) Default: FALSE
- **expandInferred** (logical) Should recognized terms in the query have inferred terms expanded? Default: TRUE
- **offset** (integer) The result (row) to start on query 0
- **count** (integer) The number of results (rows) to return. Default: 20
- **minCount** (integer) The minimum count result to return. Default: 2
- **facet** (character) Facets specified like facetName:facetValue query
- **filter** (character) Filters specified like columnName:filterValue. If the column is declared numeric then the filterValue can be a range expression (i.e.: >24.3)
- **orMultiFacets** (logical) True if multiple facet values for the same facet are specified in the query and should be ORed together. Defaults to "AND"
- **subclassFilters** (character) Filter specified like columnName:parentID where parentID is a known ID. All subclasses of parentID will be included in the filter. This parameter may be combined with "filter".
- **key** (character) API key. Required either passed in here or save as an env var or R option. See `rif` and `Startup`
- **...** Curl options passed on to `HttpClient`

See Also

Other federation: `federation_data, federation_search`

Examples

```r
# Not run:
out <- federation_facet(id = "nlx_152871-2")
out$category
out$facets

out <- federation_facet(id = "nlx_152871-2", count = 100)
out$category
out$facets

federation_facet(id = "nlx_152871-2", count = 100, minCount = 10)

# End(Not run)
```
**federation_search**  
Federation search

**Description**

Federation search

**Usage**

```r
federation_search(query, strict = FALSE, snippetCount = 0,
                   expandSynonyms = TRUE, expandAcronyms = FALSE, expandAbbrevs = FALSE,
                   expandInferred = TRUE, key = NULL, ...)
```

**Arguments**

- `query`  
  Query terms
- `strict`  
  (logical) Only consider more relevant columns. Default: FALSE
- `snippetCount`  
  How many snippets to return. Default: 0
- `expandSynonyms`  
  (logical) Should recognized terms in the query have synonyms expanded? Default: TRUE
- `expandAcronyms`  
  (logical) Default: FALSE
- `expandAbbrevs`  
  (logical) Default: FALSE
- `expandInferred`  
  (logical) Should recognized terms in the query have inferred terms expanded? Default: TRUE
- `key`  
  (character) API key. Required either passed in here or save as an env var or R option. See RIF and Startup
- `...`  
  Curl options passed on to HttpClient

**See Also**

Other federation: `federation_data`, `federation_facet`

**Examples**

```r
## Not run:
out <- federation_search("cellular")
out$\$query
out$\$messages
out$\$result$\$total
out$\$result$\$categories
head(out$\$result$\$results)

head( federation_search("cellular")$\$result$\$results )
head( federation_search("cellular", strict=TRUE)$\$result$\$results )
federation_search("cellular", snippetCount=3)
```
Lexical services - break into chunks

Usage

```r
lexical_chunks(text, key = NULL, ...)
```

Arguments

- `text` (character) The source text
- `key` (character) API key. Required either passed in here or save as an env var or R option. See `rif` and `Startup`
- `...` Curl options passed on to `crul::HttpClient()`

See Also

Other lexical: `lexical_entities, lexical_pmchunks, lexical_pmentities, lexical_pos, lexical_sentences`

Examples

```r
## Not run:
text <- "Lorem ipsum inceptos dolor nisi torquenter porttitor donec,
blandit scelerisque mattis cras quis mi, aliquam sagittis. Convallis
placerat commodo imperdiet varius nunc tempus urna vitae ultrices tristique
eu mi ornare velit donec, posuere laoreet pretium vitae porta augue porta
feugiat in sapien egestas. Quam odio nullam pulvinar litora curabitur amet
lacus sociosqu gravida ligula molestie dui netus fusce bibendum scelerisque,
dictum malesuada proin elit etiam aliquam, mattis euismod donec
nisl facilisis."

lexical_chunks(gsub("\n", "", text))
```

## End(Not run)
lexical_entities  

**Lexical services - break into entities**

**Description**

Lexical services - break into entities

**Usage**

```r
lexical_entities(text, key = NULL, ...)
```

**Arguments**

text  
(character) The source text

key  
(character) API key. required either passed in here or save as an env var or R option. see rif and Startup

...  
Curl options passed on to `crl::HttpClient()`

**See Also**

Other lexical: `lexical_chunks`, `lexical_pmchunks`, `lexical_pmentities`, `lexical_pos`, `lexical_sentences`

**Examples**

```r
## Not run:
text <- "Lorem ipsum inceptos dolor nisi torquent porttitor donec, blandit scelerisque mattis cras quis mi, aliquam sagittis. Convallis placerat commodo imperdiet varius nunc tempus urna vitae ultrices tristique eu mi ornare velit donec, posuere laoreet pretium vitae porta augue porta feugiat in sapien egestas. Quam odio nullam pulvinar litora curabitur amet lacus sociosqu gravida ligula molestie dui netus fusce bibendum scelerisque, dictum malesuada proin elit etiam aliquam, mattis euismod donec nisl facilisis."

lexical_entities(gsub("\n", "", text))

## End(Not run)
```

---

lexical_pmchunks  

**Lexical services - break Pubmed documents into chunks**

**Description**

Lexical services - break Pubmed documents into chunks
Usage

`lexical_pmentities(pmid, key = NULL, ...)`

Arguments

- `pmid` One or more PMID’s in a vector or list
- `key` (character) API key. required either passed in here or save as an env var or R option. see `rif` and `Startup`
- `...` Curl options passed on to `curl::HttpClient()`

See Also

Other lexical: `lexical_chunks, lexical_entities, lexical_pmentities, lexical_pos, lexical_sentences`

Examples

```r
## Not run:
out <- literature_retractions()
lexical_pmc_chunks(out[1:3])

## End(Not run)
```

---

**lexical_pmentities**

*Lexical services - break Pubmed documents into entities*

Description

Lexical services - break Pubmed documents into entities

Usage

`lexical_pmentities(pmid, key = NULL, ...)`

Arguments

- `pmid` One or more PMID’s in a vector or list
- `key` (character) API key. required either passed in here or save as an env var or R option. see `rif` and `Startup`
- `...` Curl options passed on to `curl::HttpClient()`

See Also

Other lexical: `lexical_chunks, lexical_entities, lexical_pmc_chunks, lexical_pos, lexical_sentences`
Lexical services - extract POS tags from input text

Usage

`lexical_pos(text, key = NULL, ...)`

Arguments

text (character) The source text
key (character) API key. required either passed in here or save as an env var or R option. see `rif` and `Startup`

... Curl options passed on to `curl::HttpClient()`

See Also

Other lexical: `lexical_chunks, lexical_entities, lexical_pmchunks, lexical_pmentities, lexical_sentences`

Examples

```r
## Not run:
text <- "Lorem ipsum inceptos dolor nisi torquent porttitor donec, blandit scelerisque mattis cras quis mi, aliquam sagittis. Convallis placerat commodo imperdiet varius nunc tempus urna vitae ultrices tristique eu mi ornare velit donec, posuere laoreet pretium vitae porta augue porta feugiat in sapien egestas. Quam odio nullam pulvinar litora curabitur amet lacus sociosqu gravida ligula molestie dui netus fusce bibendum scelerisque, dictum malesuada proin elit etiam aliquam, mattis euismod donec nisl facilisis."

lexical_pos(gsub("\n", "", text))
```

## End(Not run)
**lexical_sentences**  
*Lexical services - break into sentences*

**Description**

Lexical services - break into sentences

**Usage**

```r
lexical_sentences(text, key = NULL, ...)
```

**Arguments**

- `text` (character) The source text
- `key` (character) API key. required either passed in here or save as an env var or R option. see `rif` and `Startup`
- `...` Curl options passed on to `crl::HttpClient()`

**See Also**

Other lexical: `lexical_chunks`, `lexical_entities`, `lexical_pmchunks`, `lexical_pmentities`, `lexical_pos`

**Examples**

```r
## Not run:
text <- "Lorem ipsum inceptos dolor nisi torquent porttitor donec, blandit scelerisque mattis cras quis mi, aliquam sagittis. Convallis placerat commodo imperdiet varius nunc tempus urna vitae ultrices tristique eu mi ornare velit donec, posuere laoreet pretium vitae porta a augue porta feugiat in sapien egestas. Quam odio ullam pulvinar litora curabitur amet lacus sociosqu gravida ligula molestie dui netus fusce bibendum scelerisque, dictum malesuada proin elit etiam aliquam, mattis euismod donec nisl facilisis."

lexical_sentences(gsub("\n", "", text))
```

## End(Not run)
literature_mlpmid  

**Literature - more like PMID search**

**Description**

Literature - more like PMID search

**Usage**

```r
literature_mlpmid(pmid, offset = 0, count = 20, authorFilter = NULL,
                   yearFilter = NULL, journalFilter = NULL, key = NULL, ...)
```

**Arguments**

- **pmid**: One or more PMID’s in a vector or list
- **offset**: (integer) The result (row) to start on query 0
- **count**: (integer) The number of results (rows) to return. Default: 5
- **authorFilter**: Author filters
- **yearFilter**: Year filters
- **journalFilter**: Journal filters
- **key**: (character) API key. required either passed in here or save as an env var or R option. see rif and Startup
- **...**: Curl options passed on to `curl::HttpClient()`

**See Also**

Other literature: `literature_mltext`, `literature_pmid`, `literature_retractions`

**Examples**

```r
## Not run:
out <- literature_retractions()
literature_mlpmid(pmid = out[1:3])

literature_mlpmid(pmid = out[1:3])
literature_mlpmid(pmid = out[1:3], offset = 2)
literature_mlpmid(pmid = out[1:3], count = 3)

## End(Not run)
```
**literature_mltext**  
*Literature - more like text search*

**Description**

Literature - more like text search

**Usage**

```r
literature_mltext(text, offset = 0, count = 20, key = NULL, ...)
```

**Arguments**

- `text` (The query)
- `offset` (integer) The result (row) to start on query 0
- `count` (integer) The number of results (rows) to return. Default: 5
- `key` (character) API key. required either passed in here or save as an env var or R option. see `rif` and `Startup`
- `...` Curl options passed on to `crl::HttpClient()`

**See Also**

Other literature: *literature_mlpmid, literature_pmid, literature_retractions*

**Examples**

```r
## Not run:
literature_mltext(text = "cellular")
literature_mltext(text = "ecology", count = 200)

## End(Not run)
```

---

**literature_pmid**  
*Get articles by PMID*

**Description**

Get articles by PMID

**Usage**

```r
literature_pmid(pmids, key = NULL, ...)
```
Arguments

- **pmid**: One or more PMID’s in a vector or list
- **key**: (character) API key. required either passed in here or save as an env var or R option. see rif and Startup
- ... Curl options passed on to `crul::HttpClient()`

See Also

Other literature: `literature_mlpmid`, `literature_mltext`, `literature_retractions`

Examples

```r
## Not run:
# get PMIDs for retracted articles
(out <- literature_retractions())
# then get info on some articles
literature_pmid(pmid = out[1])
literature_pmid(pmid = out[1:2])
literature_pmid(pmid = out[1:10])
## End(Not run)
```
Examples

```r
## Not run:
# get PMIDs for retracted articles
(out <- literature_retractions())
# then get info on some articles
literature_pmid(pmid = out[1])

## End(Not run)
```

---

**literature_search**  
**Literature search**

**Description**

Literature search

**Usage**

```r
literature_search(query, offset = 0, count = 20, minDate = NULL, maxDate = NULL, facetCount = 0, snippetCount = 0, minMatch = 60, searchFullText = FALSE, requireFullText = FALSE, authorFilter = NULL, yearFilter = NULL, journalFilter = NULL, sort = NULL, section = NULL, expandSynonyms = TRUE, expandAcronyms = FALSE, expandAbbrevs = FALSE, expandInferred = TRUE, highlight = FALSE, abstract = TRUE, key = NULL, ...)
```

**Arguments**

- `query`: Query terms
- `offset`: (integer) The result (row) to start on
- `count`: (integer) The number of results (rows) to return. Default: 20
- `minDate`: The minimum date (a unix time value - milliseconds since 1970)
- `maxDate`: The maximum date (a unix time value - milliseconds since 1970)
- `facetCount`: The facet count. Default: 0
- `snippetCount`: The snippet count. Default: 0
- `minMatch`: The Solr minmatch percentage. Default: 60 percent
- `searchFullText`: Should full text be included in the search? Default: FALSE
- `requireFullText`: Should only full text be included in the search? Default: FALSE
- `authorFilter`: Author filters
- `yearFilter`: Year filters
- `journalFilter`: Journal filters
- `sort`: Use "date" to sort by date. Otherwise relevancy will be used.
section       Sections to search
expandSynonyms  (logical) Should recognized terms in the query have synonyms expanded? Default: TRUE
expandAcronyms  (logical) Default: FALSE
expandAbbrevs   (logical) Default: FALSE
expandInferred  (logical) Should recognized terms in the query have inferred terms expanded? Default: TRUE
highlight       (no documentation provided). Default: FALSE
abstract        (no documentation provided). Default: TRUE
key             (character) API key. required either passed in here or save as an env var or R option. see rif and Startup

...           Curl options passed on to HttpClient

Examples

```r
## Not run:
out <- literature_search(query = "cellular")
out$result$publications
out$result$facets
out$query

literature_search(query = "ecology")
literature_search(query = "ecology", searchFullText = FALSE)
literature_search(query = "ecology", searchFullText = TRUE)
literature_search(query = "ecology", requireFullText = TRUE)

## End(Not run)
```

---

rif_query       Query NIF data

Description

Query NIF data

Usage

```r
rif_query(query, expandSynonyms = TRUE, expandAcronyms = FALSE,
expandAbbrevs = FALSE, expandInferred = TRUE, key = NULL, ...)
```
Arguments

query Query terms
expandSynonyms (logical) Should recognized terms in the query have synonyms expanded? Default: TRUE
expandAcronyms (logical) Default: FALSE
expandAbbrevs (logical) Default: FALSE
expandInferred (logical) Should recognized terms in the query have inferred terms expanded? Default: TRUE
key (character) API key. required either passed in here or save as an env var or R option. see rif and Startup
...
Curl options passed on to HttpClient

See Also

Other query: rif_query_categories

Examples

## Not run:
rif_query(query = "anatomy")

rif_query("cellular", verbose = TRUE)

## End(Not run)

rif_query_categories Query NIF data

Description

Query NIF data

Usage

rif_query_categories(key = NULL, ...)

Arguments

key (character) API key. required either passed in here or save as an env var or R option. see rif and Startup
...
Curl options passed on to crul::HttpClient()

See Also

Other query: rif_query
Examples

```r
## Not run:
rif_query_categories()
rif_query_categories(verbos = TRUE)

## End(Not run)
```

### rif_summary

<table>
<thead>
<tr>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
</tr>
<tr>
<td>Summary</td>
</tr>
</tbody>
</table>

#### Usage

rif_summary(query, expandSynonyms = TRUE, expandAcronyms = FALSE, expandAbbrevs = FALSE, expandInferred = TRUE, key = NULL, ...)

#### Arguments

- `query` - Query terms
- `expandSynonyms` - (logical) Should recognized terms in the query have synonyms expanded? Default: TRUE
- `expandAcronyms` - (logical) Default: FALSE
- `expandAbbrevs` - (logical) Default: FALSE
- `expandInferred` - (logical) Should recognized terms in the query have inferred terms expanded? Default: TRUE
- `key` - (character) API key. required either passed in here or save as an env var or R option. see rif and Startup
- `...` - Curl options passed on to HttpClient

#### Examples

```r
## Not run:
out <- rif_summary("cellular")
out$query
out$messages
out$result$literatureSummary
out$result$federationSummary$total
out$result$federationSummary$categories
head(out$result$federationSummary$results)

head( rif_summary("neuron")$result$federationSummary$results )
head( rif_summary("neuron", expandSynonyms=FALSE)$result$federationSummary$results
```
vocabulary

```r
vocabulary(prefix = NULL, id = NULL, limit = 20, category = NULL,
vocabulary = NULL, key = NULL, ...)```

### Arguments
- `prefix`, `id` (character): The prefix, or id of the term. Use one or the other
- `limit`: The number of candidates to return. Default: 20
- `category`: A category
- `vocabulary`: A vocabulary
- `key` (character): API key. required either passed in here or save as an env var or R option. see `rif` and `Startup`
- `...`: Curl options passed on to `crul::HttpClient()`

### See Also
Other vocabulary: `vocabulary_categories, vocabulary_providers, vocabulary_search, vocabulary_suggest`

### Examples
```
## Not run:
vocabulary(prefix = "cell")
vocabulary(prefix = "cell", limit = 4)
vocabulary(prefix = "axon")
vocabulary(id = "GO_0022407")
## End(Not run)
```
vocabulary_categories  Vocabulary services - get categories

Description
Vocabulary services - get categories

Usage
vocabulary_categories(vocabulary = NULL, key = NULL, ...)

Arguments
- vocabulary: A vocabulary
- key: (character) API key. required either passed in here or save as an env var or R option. see rif and Startup
- ...: Curl options passed on to curl::HttpClient()

See Also
Other vocabulary: vocabulary_providers, vocabulary_search, vocabulary_suggest, vocabulary

Examples
```r
## Not run:
vocabulary_categories()

## End(Not run)
```

vocabulary_providers  Vocabulary services - get providers

Description
Vocabulary services - get providers

Usage
vocabulary_providers(vocabulary = NULL, key = NULL, ...)

Arguments
- vocabulary: A vocabulary
- key: (character) API key. required either passed in here or save as an env var or R option. see rif and Startup
- ...: Curl options passed on to curl::HttpClient()
vocabulary_search

See Also

Other vocabulary: vocabulary_categories, vocabulary_search, vocabulary_suggest, vocabulary

Examples

## Not run:
vocabulary_providers()

## End(Not run)

---

vocabulary_search  Vocabulary services - Search for vocabulary terms and their metadata

Description

Vocabulary services - Search for vocabulary terms and their metadata

Usage

vocabulary_search(term, limit = 20, category = NULL, vocabulary = NULL, key = NULL, ...)

Arguments

term  (character) The search terms
limit  The number of candidates to return. Default: 20
category  A category
vocabulary  A vocabulary
key  (character) API key. required either passed in here or save as an env var or R option. see rif and Startup
...

Curl options passed on to curl::HttpClient()

See Also

Other vocabulary: vocabulary_categories, vocabulary_providers, vocabulary_suggest, vocabulary

Examples

## Not run:
vocabulary_search("cell")
vocabulary_search("cell", limit = 3)
vocabulary_search("cell", category = "biological_process")

## End(Not run)
vocabulary_suggest

Description
DOESN'T SEEM TO WORK

Usage
vocabulary_suggest(query, limit = 10, key = NULL, ...)

Arguments
  query          (character) Query terms
  limit          The number of candidates to return. Default: 20
  key            (character) API key. required either passed in here or save as an env var or R
                  option. see rif and Startup
  ...            Curl options passed on to crul::HttpClient()

See Also
Other vocabulary: vocabulary_categories, vocabulary_providers, vocabulary_search, vocabulary

Examples
  ## Not run:
  vocabulary_suggest("cell")

  ## End(Not run)
Index

*Topic family
  literature_search, 15
*Topic package
  rif-package, 2

crul::HttpClient(), 7–14, 17, 19–22

federation_data, 3, 5, 6
federation_facet, 4, 4, 6
federation_search, 4, 5, 6

Http::Client, 4–6, 16–18

lexical_chunks, 7, 8–11
lexical_entities, 7, 8, 9–11
lexical_pmchunks, 7, 8, 8, 9–11
lexical_pmentities, 7–9, 9, 10, 11
lexical_pos, 7–9, 10, 11
lexical_sentences, 7–10, 11
literature_mlpmid, 12, 13, 14
literature_mltext, 12, 13, 14
literature_pmid, 12, 13, 13, 14
literature_retractions, 12–14, 14
literature_search, 15

rif, 4–14, 16–22
rif (rif-package), 2
rif-package, 2
rif_query, 16, 17
rif_query_categories, 17, 17
rif_summary, 18

Startup, 4–14, 16–22

vocabulary, 19, 20–22
vocabulary_categories, 19, 20, 21, 22
vocabulary_providers, 19, 20, 20, 21, 22
vocabulary_search, 19–21, 21, 22
vocabulary_suggest, 19–21, 22