Package 'rorcid'

September 22, 2016

Title Interface to the 'Orcid.org' 'API'

Description Client for the 'Orcid.org' 'API' (http://orcid.org/). Functions included for searching for people, searching by 'DOI', and searching by 'Orcid' 'ID'.

Version 0.3.0

License MIT + file LICENSE

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

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

LazyData true

Imports httr (>= 1.1.0), jsonlite (>= 1.0), tibble (>= 1.2)

Suggests roxygen2 (>= 5.0.1), testthat, knitr, rmarkdown

RoxygenNote 5.0.1

VignetteBuilder knitr

NeedsCompilation no

Author Scott Chamberlain [aut, cre]

Maintainer Scott Chamberlain <myrmecocystus@gmail.com>

Repository CRAN

Date/Publication 2016-09-22 01:18:06

R topics documented:

`rorcid-package` *A programmatic R interface the Orcid.org API.*

Description

A programmatic R interface the Orcid.org API.

Author(s)

Scott Chamberlain <myrmecocystus@gmail.com>

`as.orcid` *Convert an ORCID or something like an ORCID object*

Description

Convert an ORCID or something like an ORCID object

Usage

```
as.orcid(x, ...)
```

Arguments

`x`	An ORCID id, passed to `print`
`...`	Further args passed on to `orcid_id`

Value

an S3 object of class `or_cid`, which pretty prints for brevity

Examples

```
## Not run:
as.orcid(x="0000-0002-1642-628X")
out <- orcid("text:English", rows = 20)
as.orcid(out$`orcid-identifier.path`[1])

# Passon further args to orcid_id()
library("httr")
as.orcid("0000-0002-1642-628X", config=verbose())

# Browse to a profile
# browse(as.orcid("0000-0002-1642-628X"))

# many ORCIDs as a character vector
```

```
ids <- c("0000-0002-1642-628X", "0000-0002-9341-7985")
as.orcid(ids)

# many in a list via orcid_id()
(x <- orcid_id(orcid = ids))
as.orcid(x)

## End(Not run)
```

`browse` *Navigate to an ORCID profile in your default browser*

Description

Navigate to an ORCID profile in your default browser

Usage

```
browse(orcid)
```

Arguments

`orcid`	An `or_cid` class object

Examples

```
## Not run:
browse(as.orcid("0000-0002-1642-628X"))

## End(Not run)
```

`check_dois` *Verify DOI's are likely good*

Description

Verify DOI's are likely good

Usage

```
check_dois(x)
```

Arguments

`x`	One or more DOIs

Value

A list of length two, one slot for good DOIs, one for bad

Examples

```
## Not run:
check_dois("10.1087/20120404")

dois=c("10.1371/journal.pone.0025995","10.1371/journal.pone.0053712",
       "10.1371/journal.pone.0054608","10.1371/journal.pone.0055937")
check_dois(dois)

dois=c("10.1016/j.medpal.2008.12.005","10.1080/00933104.2000.10505926","10.1037/a0024480",
       "10.1002/anie.196603172","2344","asdf","232","asdf","23dd")
check_dois(dois)

## End(Not run)
```

---

fields *Lookup-table for search fields*

Description

Lookup-table for search fields

---

identifiers *Get identifiers*

Description

This function aims to pluck out just identifiers into a vector for easy use downstream (e.g., use DOIs to fetch article metadata). You can still manually fetch additional data from outputs of functions in this package.

Usage

```
identifiers(x, type = "doi", ...)

## S3 method for class 'works'
identifiers(x, type = "doi", ...)

## S3 method for class 'list'
identifiers(x, type = "doi", ...)

## S3 method for class 'orcid_id'
identifiers(x, type = "doi", ...)
```

```
## S3 method for class 'orcid'
identifiers(x, type = "doi", ...)

## S3 method for class 'orcid_doi'
identifiers(x, type = "doi", ...)
```

Arguments

x	An object of class works, orcid, orcid_id, orcid_doi, or a list that contains any number of the previous objects.
type	(character) One of doi (default), pmid, pmc, eid, other_id, orcid, scopus, researcherid. The orcid's here are for works, not individuals.
...	Ignored.

Value

A vector of identifiers, or NULL if none found

Examples

```
## Not run:
# Result of call to works()
x <- works(orcid_id("0000-0001-8607-8025"))
# doi by default
identifiers(x)
# orcids
identifiers(x, "orcid")
# pmid
identifiers(x, "pmid")
# pmc
identifiers(x, "pmc")
# other_id
identifiers(x, "other_id")

# Result of call to orcid_id()
x <- orcid_id(orcid = "0000-0002-9341-7985")
identifiers(x, "doi")
identifiers(x, "eid")

# Result of call to orcid()
x <- orcid(query="carl+boettiger")
identifiers(x, "scopus")
identifiers(x, "orcid")
identifiers(x, "researcherid")

# Result of call to orcid_doi()
x <- orcid_doi(dois="10.1087/20120404", fuzzy=TRUE)
identifiers(x, "scopus")

## End(Not run)
```

orcid *Search for ORCID ID's.*

Description

Search for ORCID ID's.

Usage

```
orcid(query = NULL, start = NULL, rows = NULL, recursive = FALSE,
  defType = NULL, q.alt = NULL, qf = NULL, mm = NULL, qs = NULL,
  pf = NULL, ps = NULL, pf2 = NULL, ps2 = NULL, pf3 = NULL,
  ps3 = NULL, tie = NULL, bq = NULL, bf = NULL, boost = NULL,
  uf = NULL, lowercaseOperators = NULL, fuzzy = FALSE, ...)
```

Arguments

query	Search terms. You can do quite complicated queries using the SOLR syntax. See examples below. For all possible fields to query, do data(fields)
start	Result number to start on. Keep in mind that pages start at 0.
rows	Numer of results to return.
recursive	Keep drilling down until all records are retrieved for the given query, default FALSE (logical). If recursive=TRUE, rows and start parameters are ignored.
defType	Query syntax. One of edismax or X. See Details for more.
q.alt	If specified, this query will be used (and parsed by default using standard query parsing syntax) when the main query string is not specified or blank. This comes in handy when you need something like a match-all-docs query (don't forget &rows=0 for that one!) in order to get collection-wise faceting counts.
qf	(Query Fields) List of fields and the "boosts" to associate with each of them when building DisjunctionMaxQueries from the user's query
mm	(Minimum 'Should' Match) See the wiki here http://wiki.apache.org/solr/ExtendedDisMax#mm_.28Minimum_.27Should.27_Match.29
qs	(Query Phrase Slop) Amount of slop on phrase queries explicitly included in the user's query string (in qf fields; affects matching).
pf	(Phrase Fields) Once the list of matching documents has been identified using the "fq" and "qf" params, the "pf" param can be used to "boost" the score of documents in cases where all of the terms in the "q" param appear in close proximity. Read more here http://wiki.apache.org/solr/ExtendedDisMax#pf_.28Phrase_Fields.29
ps	(Phrase Slop) Default amount of slop on phrase queries built with "pf", "pf2" and/or "pf3" fields (affects boosting).
pf2	(Phrase bigram fields) As with 'pf' but chops the input into bi-grams, e.g. "the brown fox jumped" is queried as "the brown" "brown fox" "fox jumped"

ps2 (Phrase bigram slop) As with 'ps' but sets default slop factor for 'pf2'. If not specified, 'ps' will be used.

pf3 (Phrase trigram fields) As with 'pf' but chops the input into tri-grams, e.g. "the brown fox jumped" is queried as "the brown fox" "brown fox jumped"

ps3 (Phrase trigram slop) As with 'ps' but sets default slop factor for 'pf3'. If not specified, 'ps' will be used.

tie (Tie breaker) Float value to use as tiebreaker in DisjunctionMaxQueries (should be something much less than 1). Read more here http://wiki.apache.org/solr/ExtendedDisMax#tie_.28Tie_breaker.29

bq (Boost Query) A raw query string (in the SolrQuerySyntax) that will be included with the user's query to influence the score. Read more here http://wiki.apache.org/solr/ExtendedDisMax#bq_.28Boost_Query.29

bf (Boost Function, additive) Functions (with optional boosts) that will be included in the user's query to influence the score. Any function supported natively by Solr can be used, along with a boost value, e.g.: recip(rord(myfield),1,2,3)^1.5. Read more here http://wiki.apache.org/solr/ExtendedDisMax#bf_.28Boost_Function.2C_additive.29

boost (Boost Function, multiplicative) As for 'bf' but multiplies the boost into the score

uf (User Fields) Specifies which schema fields the end user shall be allowed to query for explicitly. This parameter supports wildcards. Read more here http://wiki.apache.org/solr/ExtendedDisMax#uf_.28User_Fields.29

lowercaseOperators This param controls whether to try to interpret lowercase words as boolean operators such as "and", "not" and "or". Set &lowercaseOperators=true to allow this. Default is "false".

fuzzy Use fuzzy matching on input DOIs. Defaults to FALSE. If FALSE, we stick "digital-object-ids" before the DOI so that the search sent to ORCID is for that exact DOI. If TRUE, we use some regex to find the DOI.

... Curl options passed on to GET

Details

You can use any of the following within the query statement: given-names, family-name, credit-name, other-names, email, grant-number, patent-number, keyword, worktitle, digital-objectids, current-institution, affiliation-name, current-primary-institution, text, or past-institution.

For more complicated queries the ORCID API supports using ExtendedDisMax. See the documentation on the web here: http://wiki.apache.org/solr/ExtendedDisMax.

Note that when constructing queries, you don't need to use syntax like +, etc., `httr`, the curl client we use internally, will do that for you. For example, instead of writing `johnson+cardiology`, just write `johnson cardiology`, and instead of writing `johnson+AND+cardiology`, write `johnson AND cardiology`. Though, you still need to use `AND`, `OR`, etc. to join term/queries together.

See Also

`orcid_doi` `orcid_id`

Examples

```
## Not run:
# Get a list of names and Orcid IDs matching a name query
orcid(query="carl+boettiger")
orcid(query="given-names:carl AND family-name:boettiger")

# You can string together many search terms
orcid(query="johnson cardiology houston")

# And use boolean operators
orcid("johnson AND(caltech OR 'California Institute of Technology')")

# And you can use start and rows arguments to do pagination
orcid("johnson cardiology houston", start = 2, rows = 3)

# Use search terms, here family name
orcid("family-name:Sanchez", start = 4, rows = 6)

# Use search terms, here...
orcid(query="Raymond", start=0, rows=10, defType="edismax")

# Search using keywords
orcid(query="keyword:ecology")

# Search by DOI
orcid(query="10.1087/20120404")

# Note the difference between the first wrt the second and third
## See also orcid_doi() function for searching by DOIs
orcid("10.1087/20120404")
orcid('"10.1087/20120404"')
orcid('digital-object-ids:"10.1087/20120404"')

# Search by text type
orcid("text:English")

## Using more complicated SOLR queries

# Use the qf parameter to "boost" query fields so they are ranked higher
#  See how it is different than the second query without using "qf"
orcid(defType = "edismax", query = "Raymond",
   qf = "given-names^1.0 family-name^2.0", start = 0, rows = 10)
orcid(query = "Raymond", start = 0, rows = 10)

# Use other SOLR parameters as well, here mm. Using the "mm" param, 1 and
# 2 word queries require that all of the optional clauses match, but for
# queries with three or more clauses one missing clause is allowed...
# See for more: http://bit.ly/1uyMLDQ
orcid(defType = "edismax",
      query="keyword:ecology OR evolution OR conservation",
      mm = 2, rows = 20)
```

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

`orcid_doi` *Search for ORCID ID's using DOIs*

Description

Search for ORCID ID's using DOIs

Usage

```
orcid_doi(dois = NULL, start = NULL, rows = NULL, fuzzy = FALSE, ...)
```

Arguments

`dois`	(character) Digital object identifier (DOI), a vector fo DOIs.
`start`	(integer) Result number to start on. Keep in mind that pages start at 0.
`rows`	(integer) Numer of results to return.
`fuzzy`	(logical) Use fuzzy matching on input DOIs. Defaults to FALSE. If FALSE, we stick "digital-object-ids" before the DOI so that the search sent to ORCID is for that exact DOI. If TRUE, we use some regex to find the DOI.
`...`	Curl options passed on to `GET`

Examples

```
## Not run:
orcid_doi(dois="10.1087/20120404", fuzzy=TRUE)

# fuzzy is FALSE by default
orcid_doi(dois="10.1087/20120404", fuzzy=FALSE)

# This DOI is not a real one, but a partial DOI, then we can fuzzy search
# get more than defualt 10 records (or rows)
orcid_doi(dois="10.1087/2", fuzzy=TRUE, rows=20)

# If you don't input proper DOIs, the function will get mad
dois <- c("10.1371/journal.pone.0025995","10.1371/journal.pone.0053712",
       "10.1371/journal.pone.0054608","10.1371/journal.pone.0055937")
orcid_doi(dois=dois)

# dois <- c("10.1016/j.medpal.2008.12.005","10.1080/00933104.2000.10505926",
#          "10.1037/a0024480", "10.1002/anie.196603172","2344","asdf","232",
#          "asdf","23dd")
# orcid_doi(dois=dois)

orcid_doi(dois="10.1087/20120404", fuzzy=FALSE)
orcid_doi(dois="10.1371/journal.pone.0025995", fuzzy=FALSE)

## End(Not run)
```

`orcid_id` *Get data for particular ORCID's*

Description

Get data for particular ORCID's

Usage

```
orcid_id(orcid = NULL, profile = "profile", ...)
```

Arguments

`orcid`	Orcid identifier(s), of the form XXXX-XXXX-XXXX-XXXX.
`profile`	Bibliographic ("bio"), biographical ("works"), or profile ("profile"). Default: profile
`...`	Curl options passed on to `GET`

Details

This function is vectorized, so you can pass in many ORCID's, and there's an element returned for each ORCID you put in.

Value

A list of results for each Orcid ID passed in, with each element named by the Orcid ID

Examples

```
## Not run:
res <- orcid_id(orcid = "0000-0002-9341-7985")
res$`0000-0002-9341-7985`
res$`0000-0002-9341-7985`$`orcid-identifier`
res$`0000-0002-9341-7985`$`orcid-preferences`
res$`0000-0002-9341-7985`$`orcid-history`
res$`0000-0002-9341-7985`$`orcid-bio`
res$`0000-0002-9341-7985`$works

orcid_id(orcid = "0000-0002-9341-7985", "works")
orcid_id(orcid = "0000-0002-9341-7985", "bio")
orcid_id(orcid = "0000-0003-1620-1408")
orcid_id(orcid = "0000-0002-9341-7985", profile="works")
ids <- c("0000-0003-1620-1408", "0000-0002-9341-7985")
orcid_id(orcid = ids)

library("httr")
orcid_id(orcid = "0000-0003-1620-1408", config=verbose())

# only certain orcid's make employment/funding/eduction public
```

```
## some egs where it is public
### education and employment
res <- orcid_id('0000-0003-1444-9135')
res[[1]]$`orcid-activities`$affiliations$affiliation

### education, employment, and funding
res <- orcid_id('0000-0002-1642-628X')
res[[1]]$`orcid-activities`$affiliations$affiliation
res[[1]]$`orcid-activities`$`funding-list`$funding

## End(Not run)
```

`rorcid-defunct` *Defunct functions in rorcid*

Description

- `summary.or_cid`: Function is gone. Deemed not really that useful, and hard to maintain given other changes in the package.

`works` *Get works data*

Description

Get works data

Usage

```
works(x)
```

Arguments

`x`	Input from a call to `orcid_id` or `as.orcid`
`...`	Ignored.

Details

The goal of this function is to get a pretty printed quick sense of the works for 1 or more ORCID's. You can also access the complete data.frame of results. If an ORCID has works, this function prints the titles of the first 10.

Value

An S3 object of class `works`

Examples

```
## Not run:
out <- works(orcid_id("0000-0002-9341-7985"))
out
out$data

#works( orcid_id("0000-0003-1620-1408") )
works( orcid_id("0000-0002-1642-628X") )
works( orcid_id("0000-0003-1444-9135") )
works( orcid_id("0000-0003-1419-2405") )

out <- orcid(query="keyword:ecology")
works(orcid_id(out$`orcid-identifier.path`[7]))
works(orcid_id(out$`orcid-identifier.path`[8]))
works(orcid_id(out$`orcid-identifier.path`[9]))
works(orcid_id(out$`orcid-identifier.path`[10]))

works(as.orcid("0000-0002-1642-628X"))

## End(Not run)
```

Index