Package ‘ritis’

April 17, 2020

Title Integrated Taxonomic Information System Client


Version 0.9.0

License MIT + file LICENSE

URL https://github.com/ropensci/ritis (devel)
     https://docs.ropensci.org/ritis/ (docs)

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

VignetteBuilder knitr

LazyData true

Encoding UTF-8

Language en-US

Imports solrium (>= 1.1.4), curl (>= 0.9.0), jsonlite, data.table, tibble

Suggests roxygen2 (>= 7.1.0), knitr, testthat, webmockr, vcr (>= 0.5.4)

RoxygenNote 7.1.0

X-schema.org-applicationCategory Taxonomy

X-schema.org-keywords taxonomy, biology, nomenclature, JSON, API, web, api-client, identifiers, species, names

X-schema.org-isPartOf https://ropensci.org

NeedsCompilation no

Author Scott Chamberlain [aut, cre] (<https://orcid.org/0000-0003-1444-9135>), rOpenSci [fnd] (https://ropensci.org/)

Maintainer Scott Chamberlain <myrmecocystus@gmail.com>

Repository CRAN

Date/Publication 2020-04-17 18:20:06 UTC
R topics documented:

- ritis-package ................................................. 3
- accepted_names ............................................. 3
- any_match_count ........................................... 4
- comment_detail ............................................. 5
- common_names .............................................. 6
- core_metadata .............................................. 6
- coverage .................................................... 7
- credibility .................................................. 8
- currency .................................................... 9
- date_data ................................................... 9
- description ............................................... 10
- experts ..................................................... 11
- full_record ............................................... 11
- geographic_divisions .................................... 12
- geographic_values ....................................... 13
- global_species_completeness ............................. 13
- hierarchy ................................................ 14
- itis_facet ................................................ 15
- itis_group ................................................ 16
- itis_highlight ........................................... 16
- itis_search ............................................... 17
- jurisdiction .............................................. 18
- kingdoms ................................................ 19
- last_change_date ........................................ 20
- lsid2tsn ................................................... 21
- other_sources ............................................ 22
- parent_tsn ............................................... 22
- publications ............................................. 23
- rank_name ................................................ 24
- rank_names ............................................... 25
- record ................................................... 25
- review_year .............................................. 26
- scientific_name ......................................... 27
- search_anymatch ........................................ 28
- search_any_match_paged ................................. 29
- search_common ........................................... 30
- search_scientific ....................................... 31
- solr .................................................... 32
- solr_fields .............................................. 32
- synonym_names .......................................... 33
- taxon_authorship ....................................... 33
- terms ................................................... 34
- tsn2lsid .................................................. 35
- tsn_by_vernacular_language ............................ 36
- unacceptability_reason ................................. 36
- usage .................................................... 37
Description

Interface to Integrated Taxonomic Information (ITIS)

ritis package API

All functions that start with itis_ work with the ITIS Solr API described at https://www.itis.gov/solr_documentation.html, which uses the package solrium, and these functions have you use the solrium function interfaces, so you can pass on parameters to the solrium functions - so the solrium docs are important here.

All other functions work with the ITIS REST API described at https://www.itis.gov/ws_description.html. For these methods, they can grab data in either JSON or XML format. JSON is the default. We parse the JSON to R native format, either data.frame, character string, or list. You can get raw JSON as a character string back, or raw XML as a character string, and then parse yourself with jsonlite or xml2.

You’ll also be interested in the taxize book https://taxize.dev/

Terminology

- "mononomial": a taxonomic name with one part, e.g, Poa
- "binomial": a taxonomic name with two parts, e.g, Poa annua
- "trinomial": a taxonomic name with three parts, e.g, Poa annua annua

Author(s)

Scott Chamberlain <myrmecocystus@gmail.com>

Usage

accepted_names(tsn, wt = "json", raw = FALSE, ...)

Get accepted names from tsn
Arguments

- **tsn**: TSN for a taxonomic group (numeric). Required.
- **wt**: (character) One of "json" or "xml". Required.
- **raw**: (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- ... curl options passed on to `cru::HttpClient`

Value

Zero row data.frame if the name is accepted, otherwise a data.frame with information on the currently accepted name.

Examples

```r
### Not run:
# TSN accepted - good name, empty data.frame returned
accepted_names(tsn = 208527)

# TSN not accepted - input TSN is old name, non-empty data.frame returned
accepted_names(tsn = 504239)

# raw json
accepted_names(tsn = 208527, raw = TRUE)

### End(Not run)
```

---

any_match_count  

Get any match count.

Description

Get any match count.

Usage

```r
any_match_count(x, wt = "json", raw = FALSE, ...)
```

Arguments

- **x**: text or taxonomic serial number (TSN) (character or numeric)
- **wt**: (character) One of "json" or "xml". Required.
- **raw**: (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- ... curl options passed on to `cru::HttpClient`
Value

An integer containing the number of matches the search will return.

Examples

```r
## Not run:
any_match_count(x = 202385)
any_match_count(x = "dolphin")
any_match_count(x = "dolphin", wt = "xml")

## End(Not run)
```

comment_detail

Get comment detail from TSN

Description

Get comment detail from TSN

Usage

```r
comment_detail(tsn, wt = "json", raw = FALSE, ...)
```

Arguments

- **tsn**: TSN for a taxonomic group (numeric). Required.
- **wt**: (character) One of "json" or "xml". Required.
- **raw**: (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- **...**: curl options passed on to `cru::HttpClient`

Value

A data.frame with results.

Examples

```r
## Not run:
comment_detail(tsn=180543)
comment_detail(tsn=180543, wt = "xml")

## End(Not run)
```
common_names

Get common names from tsn

Description

Get common names from tsn

Usage

common_names(tsn, wt = "json", raw = FALSE, ...)

Arguments

tsn: TSN for a taxonomic group (numeric). Required.
wt: (character) One of "json" or "xml". Required.
raw: (logical) Return raw JSON or XML as character string. Required. Default: FALSE

... curl options passed on to crul::HttpClient

Value

a data.frame

Examples

## Not run:
common_names(tsn=183833)
common_names(tsn=183833, wt = "xml")

## End(Not run)

core_metadata

Get core metadata from tsn

Description

Get core metadata from tsn

Usage

core_metadata(tsn, wt = "json", raw = FALSE, ...)

## Not run:
core_metadata(tsn=183833)
core_metadata(tsn=183833, wt = "xml")

## End(Not run)
coverage

Arguments

- **tsn**: TSN for a taxonomic group (numeric). Required.
- **wt**: (character) One of "json" or "xml". Required.
- **raw**: (logical) Return raw JSON or XML as character string. Required. Default: FALSE
  ...
  curl options passed on to `curl::HttpClient`

Examples

```r
## Not run:
# coverage and currency data
core_metadata(tsn=28727)
core_metadata(tsn=28727, wt = "xml")
# no coverage or currency data
core_metadata(183671)
core_metadata(183671, wt = "xml")

## End(Not run)
```

Description

Get coverage from tsn

Usage

```r
coverage(tsn, wt = "json", raw = FALSE, ...)
```

Arguments

- **tsn**: TSN for a taxonomic group (numeric). Required.
- **wt**: (character) One of "json" or "xml". Required.
- **raw**: (logical) Return raw JSON or XML as character string. Required. Default: FALSE
  ...
  curl options passed on to `curl::HttpClient`

Examples

```r
## Not run:
# coverage data
coverage(tsn=28727)
# no coverage data
coverage(526852)
coverage(526852, wt = "xml")

## End(Not run)
```
credibility

Get credibility rating from tsn

Description

Get credibility rating from tsn

Usage

credibility_rating(tsn, wt = "json", raw = FALSE, ...)
credibility_ratings(wt = "json", raw = FALSE, ...)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tsn</td>
<td>TSN for a taxonomic group (numeric). Required.</td>
</tr>
<tr>
<td>wt</td>
<td>(character) One of &quot;json&quot; or &quot;xml&quot;. Required.</td>
</tr>
<tr>
<td>raw</td>
<td>(logical) Return raw JSON or XML as character string. Required. Default: FALSE</td>
</tr>
<tr>
<td>...</td>
<td>curl options passed on to <code>crl::HttpClient</code></td>
</tr>
</tbody>
</table>

Details

methods:

- credibility_rating: Get credibility rating for a tsn
- credibility_ratings: Get possible credibility ratings

Value

a data.frame

Examples

```r
## Not run:
credibility_rating(tsn = 526852)
credibility_rating(526852, wt = "xml")
credibility_rating(526852, raw = TRUE)

credibility_ratings()
credibility_ratings(wt = "xml")
credibility_ratings(raw = TRUE)

## End(Not run)
```
currency

Get currency from tsn

Description

Get currency from tsn

Usage

currency(tsn, wt = "json", raw = FALSE, ...)

Arguments

tsn TSN for a taxonomic group (numeric). Required.
wt (character) One of "json" or "xml". Required.
raw (logical) Return raw JSON or XML as character string. Required. Default: FALSE
...
curl options passed on to curl::HttpClient

Value

a data.frame

Examples

## Not run:
# currency data
currency(tsn=28727)
currency(tsn=28727, wt = "xml")
# no currency dat
currency(526852)
currency(526852, raw = TRUE)

## End(Not run)

date_data

Get date data from tsn

Description

Get date data from tsn

Usage

date_data(tsn, wt = "json", raw = FALSE, ...)

description

Description
Get description of the ITIS service

Usage
description(wt = "json", raw = FALSE, ...)

Arguments

wt
(character) One of "json" or "xml". Required.

raw
(logical) Return raw JSON or XML as character string. Required. Default: FALSE

... curl options passed on to crul::HttpClient

Value
a string, the ITIS web service description

Examples
## Not run:
description()
description(wt = "xml")

## End(Not run)
experts

Get expert information for the TSN.

Description
Get expert information for the TSN.

Usage

experts(tsn, wt = "json", raw = FALSE, ...)

Arguments

- `tsn` TSN for a taxonomic group (numeric). Required.
- `wt` (character) One of "json" or "xml". Required.
- `raw` (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- `...` curl options passed on to `crl::HttpClient`

Examples

## Not run:
experts(tsn = 180544)
experts(180544, wt = "xml")
experts(180544, raw = TRUE)

## End(Not run)

full_record

Get full record from TSN or lsid

Description
Get full record from TSN or lsid

Usage

full_record(tsn = NULL, lsid = NULL, wt = "json", raw = FALSE, ...)

Arguments

- `tsn` TSN for a taxonomic group (numeric). Required.
- `lsid` lsid for a taxonomic group (character)
- `wt` (character) One of "json" or "xml". Required.
- `raw` (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- `...` curl options passed on to `crl::HttpClient`
geographic_divisions

Examples

## Not run:
# from tsn
full_record(tsn = 50423)
full_record(tsn = 202385)
full_record(tsn = 183833)

full_record(tsn = 183833, wt = "xml")
full_record(tsn = 183833, raw = TRUE)

# from lsid
full_record(lsid = "urn:lsid:itis.gov:itis_tsn:180543")
full_record(lsid = "urn:lsid:itis.gov:itis_tsn:180543")

## End(Not run)

---

### geographic_divisions

*Get geographic divisions from tsn*

**Description**

Get geographic divisions from tsn

**Usage**

geographic_divisions(tsn, wt = "json", raw = FALSE, ...)

**Arguments**

- **tsn**: TSN for a taxonomic group (numeric). Required.
- **wt**: (character) One of "json" or "xml". Required.
- **raw**: (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- **...**: curl options passed on to `crl::HttpClient`

**Examples**

## Not run:
geographic_divisions(tsn = 180543)

geographic_divisions(tsn = 180543, wt = "xml")

geographic_divisions(tsn = 180543, wt = "json", raw = TRUE)

## End(Not run)
geographic_values

*Get all possible geographic values*

**Description**

Get all possible geographic values

**Usage**

```r
geographic_values(wt = "json", raw = FALSE, ...)
```

**Arguments**

- `wt` (character) One of "json" or "xml". Required.
- `raw` (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- `...` curl options passed on to `cru::HttpClient`

**Value**

character vector of geographic names

**Examples**

```r
## Not run:
geographic_values()
geographic_values(wt = "xml")
geographic_values(wt = "json", raw = TRUE)
## End(Not run)
```

---

global_species_completeness

*Get global species completeness from tsn*

**Description**

Get global species completeness from tsn

**Usage**

```r
global_species_completeness(tsn, wt = "json", raw = FALSE, ...)
```
hierarchy

Arguments

- **tsn**: TSN for a taxonomic group (numeric). Required.
- **wt**: (character) One of "json" or "xml". Required.
- **raw**: (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- ... curl options passed on to crul::HttpClient

Examples

```r
## Not run:
global_species_completeness(tsn = 180541)
global_species_completeness(180541, wt = "xml")
global_species_completeness(180541, wt = "json", raw = TRUE)
## End(Not run)
```

hierarchy

Get hierarchy down from tsn

Description

Get hierarchy down from tsn

Usage

- `hierarchy_down(tsn, wt = "json", raw = FALSE, ...)`
- `hierarchy_up(tsn, wt = "json", raw = FALSE, ...)`
- `hierarchy_full(tsn, wt = "json", raw = FALSE, ...)`

Arguments

- **tsn**: TSN for a taxonomic group (numeric). Required.
- **wt**: (character) One of "json" or "xml". Required.
- **raw**: (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- ... curl options passed on to crul::HttpClient

Details

Hierarchy methods:
- `hierarchy_down`: Get hierarchy down from tsn
- `hierarchy_up`: Get hierarchy up from tsn
- `hierarchy_full`: Get full hierarchy from tsn
Examples

```r
## Not run:
## Full down (class Mammalia)
hierarchy_down(tsn=179913)

## Full up (genus Agoseris)
hierarchy_up(tsn=36485)

## Full hierarchy
### genus Liatris
hierarchy_full(tsn=37906)
### get raw data back
hierarchy_full(tsn=37906, raw = TRUE)
### genus Baetis, get xml back
hierarchy_full(100800, wt = "xml")

## End(Not run)
```

itis_facet  ITIS Solr facet

Description

ITIS Solr facet

Usage

```r
itis_facet(..., proxy = NULL, callopts = list())
```

Arguments

- `...` Arguments passed on to the `params` parameter of the `solrium::solr_facet()` function. See `solr_fields` for possible parameters, and examples below
- `proxy` List of arguments for a proxy connection, including one or more of: url, port, username, password, and auth. See `crul::proxy()` for help, which is used to construct the proxy connection.
- `callopts` Curl options passed on to `crul::HttpClient`

Examples

```r
## Not run:
itis_facet(q = "rank:Species", rows = 0, facet.field = "kingdom")$facet_fields

x <- itis_facet(q = "hierarchySoFar:*$Aves$* AND rank:Species AND usage:valid", facet.pivot = "nameWInd,vernacular", facet.limit = -1, facet.mincount = 1, rows = 0)
head(x$facet_pivot$\'nameWInd,vernacular\')

## End(Not run)
```
itis_group

**ITIS Solr group search**

**Description**

ITIS Solr group search

**Usage**

```r
itis_group(..., proxy = NULL, callopts = list())
```

**Arguments**

- `...`: Arguments passed on to the `params` parameter of the `solrium::solr_group()` function. See `solr_fields` for possible parameters, and examples below
- `proxy`: List of arguments for a proxy connection, including one or more of: url, port, username, password, and auth. See `cru::proxy()` for help, which is used to construct the proxy connection.
- `callopts`: Curl options passed on to `cru::HttpClient`

**Examples**

```r
## Not run:
x <- itis_group(q = "nameWOInd:/[A-Za-z0-9]*%20{0,0}*/", 
group.field = 'rank', group.limit = 3)
head(x)

## End(Not run)
```

itis_highlight

**ITIS Solr highlight**

**Description**

ITIS Solr highlight

**Usage**

```r
itis_highlight(..., proxy = NULL, callopts = list())
```

**Arguments**

- `...`: Arguments passed on to the `params` parameter of the `solrium::solr_highlight()` function. See `solr_fields` for possible parameters, and examples below
- `proxy`: List of arguments for a proxy connection, including one or more of: url, port, username, password, and auth. See `cru::proxy()` for help, which is used to construct the proxy connection.
- `callopts`: Curl options passed on to `cru::HttpClient`
### Examples

```r
## Not run:
itis_highlight(q = "rank:Species", hl.fl = 'rank', rows=10)

## End(Not run)
```

#### Description

ITIS Solr search

#### Usage

```r
itis_search(..., proxy = NULL, callopts = list())
```

#### Arguments

- `...`: Arguments passed on to the `params` parameter of the `solr::solr_search()` function. See `solr_fields` for possible parameters, and examples below
- `proxy`: List of arguments for a proxy connection, including one or more of: url, port, username, password, and auth. See `crul::proxy()` for help, which is used to construct the proxy connection.
- `callopts`: Curl options passed on to `crul::HttpClient`

#### Details

The syntax for this function can be a bit hard to grasp. See https://itis.gov/solr_examples.html for help on generating the syntax ITIS wants for specific searches.

#### References

- [https://www.itis.gov/solr_documentation.html](https://www.itis.gov/solr_documentation.html)

#### Examples

```r
## Not run:
itis_search(q = "tsn:182662")
```

# get all orders within class Aves (birds)
```r
z <- itis_search(q = "rank:Class AND nameWOInd:Aves")
hierarchy_down(z$tsn)
```

# get taxa "downstream" from a target taxon
```r
# taxize and taxizedb packages have downstream() fxns, but
# you can do a similar thing here by iteratively drilling down
# the taxonomic hierarchy
```
## here, we get families within Aves
library(data.table)
aves <- itis_search(q = "rank:Class AND nameWOInd:Aves")
aves_orders <- hierarchy_down(aves$tsn)
aves_families <- lapply(aves_orders$tsn, hierarchy_down)
rbindlist(aves_families)

# the tila operator
itis_search(q = "nameWOInd:Liquidambar\ styraciflua~0.4")

# matches only monomials
itis_search(q = "nameWOInd:/[A-Za-z0-9]*[ ]{0,0}*/")

# matches only binomials
itis_search(q = "nameWOInd:/[A-Za-z0-9]*[ ]{1,1}[A-Za-z0-9]*/*/")

# matches only trinomials
itis_search(q = "nameWOInd:/[A-Za-z0-9]*[ ]{1,1}[A-Za-z0-9]*[ ]{1,1}[A-Za-z0-9]*/*/")

# matches binomials or trinomials
itis_search(q = "nameWOInd:/[A-Za-z0-9]*[ ]{1,1}[A-Za-z0-9]*[ ]{0,1}[A-Za-z0-9]*/*/")

itis_search(q = "nameWOInd:Poa\ annua")

# pagination
itis_search(q = "nameWOInd:/[A-Za-z0-9]*[ ]{0,0}*/", rows = 2)
itis_search(q = "nameWOInd:/[A-Za-z0-9]*[ ]{0,0}*/", rows = 200)

# select fields to return
itis_search(q = "nameWOInd:/[A-Za-z0-9]*[ ]{0,0}*/",
    fl = c('nameWOInd', 'tsn'))

## End(Not run)

---

### jurisdiction

Get jurisdictional origin from tsn

## Description

Get jurisdictional origin from tsn

## Usage

jurisdictional_origin(tsn, wt = "json", raw = FALSE, ...)

jurisdiction_origin_values(wt = "json", raw = FALSE, ...)

jurisdiction_values(wt = "json", raw = FALSE, ...)

kingdoms 19

Arguments

- tsn (numeric) TSN for a taxonomic group. Required.
- wt (character) One of "json" or "xml". Required.
- raw (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- ... curl options passed on to curl::HttpClient

Details

Jurisdiction methods:

- jurisdictional_origin: Get jurisdictional origin from tsn
- jurisdiction_origin_values: Get jurisdiction origin values
- jurisdiction_values: Get all possible jurisdiction values

Value

- jurisdictional_origin: data.frame
- jurisdiction_origin_values: data.frame
- jurisdiction_values: character vector

Examples

```r
## Not run:
jurisdictional_origin(tsn=180543)
jurisdictional_origin(tsn=180543, wt = "xml")

jurisdiction_origin_values()
jurisdiction_values()
## End(Not run)
```

---

kingdoms  Get kingdom names from tsn

Description

Get kingdom names from tsn

Usage

```r
kingdom_name(tsn, wt = "json", raw = FALSE, ...)
kingdom_names(wt = "json", raw = FALSE, ...)
```
Arguments

- `tsn` (numeric): TSN for a taxonomic group. Required.
- `wt` (character): One of "json" or "xml". Required.
- `raw` (logical): Return raw JSON or XML as character string. Required. Default: FALSE
- ... curl options passed on to `crl::HttpClient`

Details

- `kingdom_name`: Get kingdom name for a TSN
- `kingdom_names`: Get all possible kingdom names

Examples

```r
## Not run:
kingdom_name(202385)
kingdom_name(202385, wt = "xml")
kingdom_names()
## End(Not run)
```

Description

Provides the date the ITIS database was last updated

Usage

```r
last_change_date(wt = "json", raw = FALSE, ...)
```

Arguments

- `wt` (character): One of "json" or "xml". Required.
- `raw` (logical): Return raw JSON or XML as character string. Required. Default: FALSE
- ... curl options passed on to `crl::HttpClient`

Value

character value with a date
**Examples**

```r
## Not run:
last_change_date()
last_change_date(wt = "xml")

## End(Not run)
```

---

### lsid2tsn

*Gets the TSN corresponding to the LSID, or an empty result if there is no match.*

**Description**

Gets the TSN corresponding to the LSID, or an empty result if there is no match.

**Usage**

```r
lsid2tsn(lsid, wt = "json", raw = FALSE, ...)
```

**Arguments**

- `lsid` (character) Lsid for a taxonomic group. Required.
- `wt` (character) One of "json" or "xml". Required.
- `raw` (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- `...` curl options passed on to `crl::HttpClient`

**Examples**

```r
## Not run:
lsid2tsn(lsid="urn:lsid:itis.gov:itis_tsn:28726")
lsid2tsn(lsid="urn:lsid:itis.gov:itis_tsn:28726", wt = "xml")
lsid2tsn("urn:lsid:itis.gov:itis_tsn:0")
lsid2tsn("urn:lsid:itis.gov:itis_tsn:0", wt = "xml")

## End(Not run)
```
other_sources  Returns a list of the other sources used for the TSN.

Description
Returns a list of the other sources used for the TSN.

Usage
other_sources(tsn, wt = "json", raw = FALSE, ...)

Arguments
- tsn: TSN for a taxonomic group (numeric). Required.
- wt: (character) One of "json" or "xml". Required.
- raw: (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- ...: curl options passed on to curl::HttpClient

Examples
```r
## Not run:
# results
other_sources(tsn=182662)
# no results
other_sources(tsn=2085272)
# get xml
other_sources(tsn=182662, wt = "xml")
## End(Not run)
```

parent_tsn  Returns the parent TSN for the entered TSN.

Description
Returns the parent TSN for the entered TSN.

Usage
parent_tsn(tsn, wt = "json", raw = FALSE, ...)
publications

Arguments

tsn TSN for a taxonomic group (numeric). Required.
wt (character) One of "json" or "xml". Required.
raw (logical) Return raw JSON or XML as character string. Required. Default: FALSE.

... curl options passed on to `cru::HttpClient`

Value

a data.frame

Examples

```r
## Not run:
parent_tsn(tsn = 202385)
parent_tsn(tsn = 202385, raw = TRUE)
parent_tsn(tsn = 202385, wt = "xml")

## End(Not run)
```

publication Returns a list of the publications used for the TSN.

Description

Returns a list of the publications used for the TSN.

Usage

`publications(tsn, wt = "json", raw = FALSE, ...)`

Arguments

tsn TSN for a taxonomic group (numeric). Required.
wt (character) One of "json" or "xml". Required.
raw (logical) Return raw JSON or XML as character string. Required. Default: FALSE.

... curl options passed on to `cru::HttpClient`

Value

a data.frame
rank_name

Examples

## Not run:
publications(tsn = 70340)
publications(tsn = 70340, wt = "xml")

publications(tsn = 70340, verbose = TRUE)

## End(Not run)

---

**rank_name**

*Returns the kingdom and rank information for the TSN.*

Description

Returns the kingdom and rank information for the TSN.

Usage

`rank_name(tsn, wt = "json", raw = FALSE, ...)`

Arguments

- `tsn` TSN for a taxonomic group (numeric). Required.
- `wt` (character) One of "json" or "xml". Required.
- `raw` (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- `...` curl options passed on to `cru::HttpClient`

Value

a data.frame, with rank name and other info

Examples

## Not run:
rank_name(tsn = 202385)

## End(Not run)
Description

Provides a list of all the unique rank names contained in the database and their kingdom and rank ID values.

Usage

`rank_names(wt = "json", raw = FALSE, ...)`

Arguments

- `wt` (character) One of "json" or "xml". Required.
- `raw` (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- `...` curl options passed on to `curl::HttpClient`

Value

a data.frame, with columns:

- kingdomname
- rankid
- rankname

Examples

```r
## Not run:
rank_names()
## End(Not run)
```

Description

Gets a record from an LSID

Usage

`record(lsid, wt = "json", raw = FALSE, ...)"
review_year

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lsid</td>
<td>lsid for a taxonomic group (character). Required.</td>
</tr>
<tr>
<td>wt</td>
<td>(character) One of &quot;json&quot; or &quot;xml&quot;. Required.</td>
</tr>
<tr>
<td>raw</td>
<td>(logical) Return raw JSON or XML as character string. Required. Default: FALSE</td>
</tr>
<tr>
<td>...</td>
<td>curl options passed on to <code>cru::HttpClient</code></td>
</tr>
</tbody>
</table>

Details

Gets the partial ITIS record for the TSN in the LSID, found by comparing the TSN in the search key to the TSN field. Returns an empty result set if there is no match or the TSN is invalid.

Value

a data.frame

Examples

```r
## Not run:
record(lsid = "urn:lsid:itis.gov:itis_tsn:180543")
## End(Not run)
```

---

review_year | Returns the review year for the TSN.

Description

Returns the review year for the TSN.

Usage

`review_year(tsn, wt = "json", raw = FALSE, ...)`

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tsn</td>
<td>TSN for a taxonomic group (numeric). Required.</td>
</tr>
<tr>
<td>wt</td>
<td>(character) One of &quot;json&quot; or &quot;xml&quot;. Required.</td>
</tr>
<tr>
<td>raw</td>
<td>(logical) Return raw JSON or XML as character string. Required. Default: FALSE</td>
</tr>
<tr>
<td>...</td>
<td>curl options passed on to <code>cru::HttpClient</code></td>
</tr>
</tbody>
</table>

Value

a data.frame
scientific_name

Examples

```r
## Not run:
review_year(tsn = 180541)

## End(Not run)
```

<table>
<thead>
<tr>
<th>scientific_name</th>
<th>Returns the scientific name for the TSN. Also returns the component parts (names and indicators) of the scientific name.</th>
</tr>
</thead>
</table>

Description

Returns the scientific name for the TSN. Also returns the component parts (names and indicators) of the scientific name.

Usage

```r
scientific_name(tsn, wt = "json", raw = FALSE, ...)
```

Arguments

- `tsn` TSN for a taxonomic group (numeric). Required.
- `wt` (character) One of "json" or "xml". Required.
- `raw` (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- `...` curl options passed on to `crl::HttpClient`

Value

a data.frame

Examples

```r
## Not run:
scientific_name(tsn = 531894)

## End(Not run)
```
search_anymatch  

Search for any match

Description

Search for any match

Usage

search_anymatch(x, wt = "json", raw = FALSE, ...)

Arguments

x  
text or taxonomic serial number (TSN) (character or numeric)

wt  
(character) One of "json" or "xml". Required.

raw  
(logical) Return raw JSON or XML as character string. Required. Default: FALSE

...  
curl options passed on to crul::HttpClient

Value

a data.frame

See Also

search_any_match_paged

Examples

```r
## Not run:
search_anymatch(x = 202385)
search_anymatch(x = "dolphin")
# no results
search_anymatch(x = "Pisces")

## End(Not run)
```
search_any_match_paged

Search for any matched page

Description

Search for any matched page

Usage

```r
search_any_match_paged(
  x,
  pagesize = NULL,
  pagenum = NULL,
  ascend = NULL,
  wt = "json",
  raw = FALSE,
  ...
)
```

Arguments

- `x`: text or taxonomic serial number (TSN) (character or numeric)
- `pagesize`: An integer containing the page size (numeric)
- `pagenum`: An integer containing the page number (numeric)
- `ascend`: A boolean containing true for ascending sort order or false for descending (logical)
- `wt`: (character) One of "json" or "xml". Required.
- `raw`: (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- `...`: curl options passed on to `curl::HttpClient`

Value

- a data.frame
- a data.frame

See Also

- `search_anymatch`
Examples

## Not run:
search_any_match_paged(x=202385, pagesize=100, pagenum=1, ascend=FALSE)
search_any_match_paged(x="Zy", pagesize=100, pagenum=1, ascend=FALSE)

## End(Not run)

---

**search_common**

*Search for tsn by common name*

Description

Search for tsn by common name

Usage

search_common(x, from = "all", wt = "json", raw = FALSE, ...)

Arguments

- **x**: text or taxonomic serial number (TSN) (character or numeric)
- **from**: (character) One of "all", "begin", or "end". See Details.
- **wt**: (character) One of "json" or "xml". Required.
- **raw**: (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- **...**: curl options passed on to `crl::HttpClient`

Details

The `from` parameter:

- **all**: Search against the searchByCommonName API route, which searches entire name string
- **begin**: Search against the searchByCommonNameBeginsWith API route, which searches for a match at the beginning of a name string
- **end**: Search against the searchByCommonNameEndsWith API route, which searches for a match at the end of a name string

Value

a data.frame

See Also

`search_scientific()`
## Examples

```r
## Not run:
search_common("american bullfrog")
search_common("ferret-badger")
search_common("polar bear")

# comparison: all, begin, end
search_common("inch")
search_common("inch", from = "begin")
search_common("inch", from = "end")

# end
search_common("snake", from = "end")

## End(Not run)
```

---

**search_scientific**  
*Search by scientific name*

### Description

Search by scientific name

### Usage

```r
search_scientific(x, wt = "json", raw = FALSE, ...)
```

### Arguments

- `x`  
text or taxonomic serial number (TSN) (character or numeric)

- `wt`  
(character) One of "json" or "xml". Required.

- `raw`  
(logical) Return raw JSON or XML as character string. Required. Default: FALSE

- `...`  
curl options passed on to `curl::HttpClient`

### Value

a data.frame

### See Also

`search_common`
Examples

```r
## Not run:
solr_search_scientific("Tardigrada")
solr_search_scientific("Quercus douglasii")
## End(Not run)
```

**solr**  

**ITIS Solr Methods**

**Description**

ITIS provides access to their data via their Solr service described at [https://www.itis.gov/solr_documentation.html](https://www.itis.gov/solr_documentation.html). This is a powerful interface to ITIS data as you have access to a very flexible query interface.

**Details**

See `solr_fields` and [https://www.itis.gov/solr_documentation.html](https://www.itis.gov/solr_documentation.html) for guidance on available fields.

**Functions**

- `itis_search()` - Search
- `itis_group()` - Group
- `itis_highlight()` - Highlight
- `itis_facet()` - Facet

**solr_fields**  

**List of fields that can be used in solr functions**

**Description**

Each element in the list has a list of length tree, with:

**Format**

A list of length 36

**Details**

- field: the field name, this is the name you can use in your queries
- definition: the definition of the field
- example: an example value

**Source**

[https://www.itis.gov/solr_documentation.html](https://www.itis.gov/solr_documentation.html)
synonym_names

Returns a list of the synonyms (if any) for the TSN.

Description

Returns a list of the synonyms (if any) for the TSN.

Usage

synonym_names(tsn, wt = "json", raw = FALSE, ...)

Arguments

- tsn: TSN for a taxonomic group (numeric). Required.
- wt: (character) One of "json" or "xml". Required.
- raw: (logical) Return raw JSON or XML as character string. Required. Default: FALSE.
- ...: curl options passed on to `curl::HttpClient`

Value

a data.frame

Examples

```r
## Not run:
synonym_names(tsn=183671) # tsn not accepted
synonym_names(tsn=526852) # tsn accepted

## End(Not run)
```

taxon_authorship

Returns the author information for the TSN.

Description

Returns the author information for the TSN.

Usage

taxon_authorship(tsn, wt = "json", raw = FALSE, ...)

terms

Arguments

- **tsn**: TSN for a taxonomic group (numeric). Required.
- **wt**: (character) One of "json" or "xml". Required.
- **raw**: (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- ... curl options passed on to `cru::HttpClient`

Value

a data.frame

Examples

```r
## Not run:
taxon_authorship(tsn = 183671)
## End(Not run)
```

Description

Get ITIS terms, i.e., tsn’s, authors, common names, and scientific names.

Usage

terms(query, what = "both", wt = "json", raw = FALSE, ...)

Arguments

- **query**: One or more common or scientific names, or partial names
- **what**: One of both (search common and scientific names), common (search just common names), or scientific (search just scientific names)
- **wt**: (character) One of "json" or "xml". Required.
- **raw**: (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- ... curl options passed on to `cru::HttpClient`
Examples

## Not run:
# Get terms searching both common and scientific names
terms(query='bear')

# Get terms searching just common names
terms(query='tarweed', "common")

# Get terms searching just scientific names
terms(query='Poa annua', "scientific")

# many at once
terms(query=c('Poa annua', 'Pinus contorta'), "scientific")

## End(Not run)

tsn2lsid

*Gets the unique LSID for the TSN, or an empty result if there is no match.*

Description

Gets the unique LSID for the TSN, or an empty result if there is no match.

Usage

tsn2lsid(tsn, wt = "json", raw = FALSE, ...)

Arguments

tsn  TSN for a taxonomic group (numeric). Required.
wt   (character) One of "json" or "xml". Required.
raw  (logical) Return raw JSON or XML as character string. Required. Default: FALSE
...  curl options passed on to `cru::HttpClient`

Value

a character string, an LSID, or NULL if nothing found

Examples

## Not run:
# Get terms searching both common and scientific names
terms(query='bear')

# Get terms searching just common names
terms(query='tarweed', "common")

# Get terms searching just scientific names
terms(query='Poa annua', "scientific")

# many at once
terms(query=c('Poa annua', 'Pinus contorta'), "scientific")

## End(Not run)
tsn_by_vernacular_language

Get tsn by vernacular language

Description

Get tsn by vernacular language

Usage

tsn_by_vernacular_language(language, wt = "json", raw = FALSE, ...)

Arguments

language A string containing the language. This is a language string, not the international language code (character)
wt (character) One of "json" or "xml". Required.
raw (logical) Return raw JSON or XML as character string. Required. Default: FALSE
...
curl options passed on to curl::HttpClient

Value

a data.frame

Examples

## Not run:
  tsn_by_vernacular_language(language = "french")

## End(Not run)

unacceptability_reason

Returns the unacceptability reason, if any, for the TSN.

Description

Returns the unacceptability reason, if any, for the TSN.

Usage

unacceptability_reason(tsn, wt = "json", raw = FALSE, ...)
usage

Arguments

- **tsn**: TSN for a taxonomic group (numeric). Required.
- **wt**: (character) One of "json" or "xml". Required.
- **raw**: (logical) Return raw JSON or XML as character string. Required. Default: FALSE

... curl options passed on to \texttt{curl::HttpClient}

Examples

```r
## Not run:
unacceptability_reason(tsn = 183671)

## End(Not run)
```

usage

\textit{Returns the usage information for the TSN.}

Description

Returns the usage information for the TSN.

Usage

\texttt{usage(tsn, wt = \texttt{"json"}, raw = FALSE, ...)}

Arguments

- **tsn**: TSN for a taxonomic group (numeric). Required.
- **wt**: (character) One of "json" or "xml". Required.
- **raw**: (logical) Return raw JSON or XML as character string. Required. Default: FALSE

... curl options passed on to \texttt{curl::HttpClient}

Examples

```r
## Not run:
usage(tsn = 526852)
usage(tsn = 526852, raw = TRUE)
usage(tsn = 526852, wt = \texttt{"xml"})

## End(Not run)
```
Description

Provides a list of the unique languages used in the vernacular table.

Usage

vernacular_languages(wt = "json", raw = FALSE, ...)

Arguments

wt (character) One of "json" or "xml". Required.
raw (logical) Return raw JSON or XML as character string. Required. Default: FALSE
...
   curl options passed on to curl::HttpClient

Value

a character vector of vernacular names

Examples

## Not run:
vernacular_languages()

## End(Not run)
Index

*Topic data
  solr_fields, 32

accepted_names, 3
any_match_count, 4

comment_detail, 5
common_names, 6
core_metadata, 6
coverage, 7
credibility, 8
credibility_rating (credibility), 8
credibility_ratings (credibility), 8
crul::HttpClient, 4–17, 19–31, 33–38
crul::proxy(), 15–17
currency, 9
date_data, 9
description, 10
experts, 11
full_record, 11
geographic_divisions, 12
global_species_completeness, 13
hierarchy, 14
driver (hierarchy), 14
driver (hierarchy), 14
hierarchy_up (hierarchy), 14

itis_facet, 15
itis_facet(), 32
itis_group, 16
itis_group(), 32
itis_highlight, 16
itis_highlight(), 32
itis_search, 17
itis_search(), 32
jurisdiction, 18
jurisdiction_origin_values
  (jurisdiction), 18
jurisdiction_values (jurisdiction), 18
jurisdictional_origin (jurisdiction), 18

kingdom_name (kingdoms), 19
kingdom_names (kingdoms), 19
kingdoms, 19

last_change_date, 20
lsid2tsn, 21

other_sources, 22
parent_tsn, 22
publications, 23

rank_name, 24
rank_names, 25
record, 25
review_year, 26

ritis (ritis-package), 3
ritis-package, 3

scientific_name, 27

search_any_match_paged, 28, 29
search_anymatch, 28, 29
search_common, 30, 31
search_scientific, 31
search_scientific(), 30
solr, 32, 32
solr_fields, 15–17, 32, 32
solrium::solr_facet(), 15
solrium::solr_group(), 16
solrium::solr_highlight(), 16
solrium::solr_search(), 17

synonym_names, 33
taxon_authorship, 33
terms, 34
tsn2lsid, 35
tsn_by_vernacular_language, 36
unacceptability_reason, 36
usage, 37
vernacular_languages, 38