Package 'originr'

May 1, 2018

Type Package

Title Fetch Species Origin Data from the Web

Description Get species origin data (whether species is native/invasive) from the following sources on the web: Encyclopedia of Life (<http://eol.org>), Flora 'Europaea' (<http://rbg-web2.rbge.org.uk/FE/fe.html>), Global Invasive Species Database (<http://www.iucngisd.org/gisd>), the Native Species Resolver (<http://bien.nceas.ucsb.edu/bien/tools/nsr/nsr-ws/>), Integrated Taxonomic Information Service (<http://www.itis.gov/>), and Global Register of Introduced and Invasive Species (<http://www.griis.org/>).

Version 0.3.0

License MIT + file LICENSE

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

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

LazyLoad yes

LazyData yes

Imports crul (>= 0.5.2), jsonlite (>= 1.5), data.table, xml2, taxize (>= 0.9.0)

Suggests testthat, roxygen2 (>= 6.0.1)

RoxygenNote 6.0.1

X-schema.org-applicationCategory Biology

X-schema.org-keywords species, native, invasive, origin, web, API, eol, gisd, nsr, itis, griis

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

NeedsCompilation no

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

Maintainer Scott Chamberlain <myrmecocystus@gmail.com>

Repository CRAN

Date/Publication 2018-04-30 22:05:26 UTC

R topics documented:

`originr-package` *originr - Species Origin Data*

Description

originr - Species Origin Data

Data sources in the package

- Encyclopedia of Life (http://eol.org)
- Flora Europaea (http://rbg-web2.rbge.org.uk/FE/fe.html)
- Global Invasive Species Database (http://www.iucngisd.org/gisd)
- Native Species Resolver (http://bien.nceas.ucsb.edu/bien/tools/nsr/nsr-ws/)
- Integrated Taxonomic Information Service (http://www.itis.gov/)

Author(s)

Scott Chamberlain <myrmecocystus@gmail.com>

Ignasi Bartomeus <nacho.bartomeus@gmail.com>

`eol` *Search for presence of taxonomic names in EOL invasive species databases.*

Description

See Details for important information.

Usage

```
eol(name, dataset = "all", searchby = grep, page = NULL, per_page = 500,
  key = NULL, messages = TRUE, count = FALSE, ...)

eol_invasive_data(...)
```

Arguments

name	A taxonomic name, or a vector of names.
dataset	One of all, gisd100, gisd, isc, daisie, i3n, or mineps. See the Details for what each dataset ID.
searchby	One of 'grep' (exact match) or 'agrep' (fuzzy match)
page	A maximum of 30 results are returned per page. This parameter allows you to fetch more pages of results if there are more than 30 matches (Default: 1)
per_page	Results to get per page. Default: 500
key	Your EOL API key; loads from .Rprofile.
messages	(logical) If TRUE the actual taxon queried is printed on the console.
count	(logical) If TRUE, give back a count of number of taxa listed as invasive, if FALSE (default), the normal output is given.
...	curl options passed on to HttpClient

Details

`eol_invasive_data()` gives you the entire data.frame from the "dataset=all", while `eol()` let's you search on a vector of names against any of the datasets

IMPORTANT: note that setting `dataset="all"` will give you surprising results. EOL does not include informaiton on which of the invasive datasets (i.e., gisd100, gisd, isc, daisie, i3n, or mineps) the taxon is found in, and sometimes e.g., if taxon X is in GISD, you might not find it in "all", weird. I don't know why that's happening, but it shouldn't happen.

IMPORTANT: When you get a returned NaN for a taxon, that means it's not on the invasive list in question. If the taxon is found, a taxon identifier is returned.

Beware that some datasets are quite large, and may take 30 sec to a minute to pull down all data before we can search for your species. Note there is no parameter in this API method for searching by taxon name.

`eol()` is vectorized, so you can pass a single name or a vector of names.

It's possible to return JSON or XML with the EOL API. However, this function only returns JSON.

Options for the dataset parameter are

- all - All datasets
- gisd100 - 100 of the World's Worst Invasive Alien Species (Global Invasive Species Database) http://eol.org/collections/54500
- gisd - Global Invasive Species Database 2013 http://eol.org/collections/54983
- isc - Centre for Agriculture and Biosciences International Invasive Species Compendium (ISC) http://eol.org/collections/55180

- daisie - Delivering Alien Invasive Species Inventories for Europe (DAISIE) Species List http://eol.org/collections/55179
- i3n - IABIN Invasives Information Network (I3N) Species http://eol.org/collections/55176
- mineps - Marine Invaders of the NE Pacific Species http://eol.org/collections/55331

Datasets are not updated that often. Here's last updated dates for some of the datasets as of 2014-08-25

- gisd100 updated 6 mos ago
- gisd updated 1 yr ago
- isc updated 1 yr ago
- daisie updated 1 yr ago
- i3n updated 1 yr ago
- mineps updated 1 yr ago

Value

A list of data.frame's/strings with results, with each element named by the input elements to the name parameter.

References

See info for each data source at http://eol.org/collections/55367/taxa

Examples

```
## Not run:
eol(name='Brassica oleracea', dataset='gisd')
eol(name=c('Lymantria dispar','Cygnus olor','Hydrilla verticillata',
  'Pinus concolor'), dataset='gisd')
eol(name='Sargassum', dataset='gisd')
eol(name='Ciona intestinalis', dataset='mineps')
eol(name=c('Lymantria dispar','Cygnus olor','Hydrilla verticillata',
  'Pinus concolor'), dataset='i3n')
eol(name=c('Branta canadensis','Gallus gallus','Myiopsitta monachus'),
   dataset='daisie')
eol(name=c('Branta canadensis','Gallus gallus','Myiopsitta monachus'),
  dataset='isc')

# Count
eol(name=c('Lymantria dispar','Cygnus olor','Hydrilla verticillata',
  'Pinus concolor'), dataset='gisd', count = TRUE)

# curl options
eol(name='Sargassum', dataset='gisd', verbose = TRUE)

## End(Not run)
```

flora_europaea *Check species status (native/exotic) in Flora Europaea*

Description

This function check the status (native or exotic) of a species in each of the eu countries.

For that end, it checks Flora Europaea (http://rbg-web2.rbge.org.uk/FE/fe.html) and scrapes the data from there.

Note that the webpage contains more information.

As expected, the function is as good as the database is. I think for native species is robust but new exotic species are not added as to my knowledge the database is not updated anymore. The database is not able to recognize species synonyms.

See http://rbg-web2.rbge.org.uk/FE/data/countries for explanation of the database codes.

Usage

```
flora_europaea(sp, messages = TRUE, ...)
```

Arguments

sp	character; a vector of length one with a single scientific species names in the form of c("Genus species").
messages	logical; If TRUE (default), informative messages printed
...	curl options passed on to HttpClient

Value

A list of vectors containing the countries where the species is native, exotic, ...

Author(s)

Ignasi Bartomeus <nacho.bartomeus@gmail.com>

Examples

```
## Not run:
sp <- c("Lavandula stoechas", "Carpobrotus edulis", "Rhododendron ponticum",
       "Alkanna lutea", "Anchusa arvensis")
flora_europaea(sp[1])
sapply(sp, flora_europaea, simplify = FALSE)

flora_europaea('Calendula officinalis')

## End(Not run)
```

gisd *Check invasive species status for a set of species from GISD database*

Description

This function check which species (both plants and animals) are considered "invaders" somewhere in the world.

For that end, it checks GISD (http://www.iucngisd.org/gisd) and returns a value, either "Not in GISD" or the brief description presented in GISD.

Note that the webpage contains more information. Also note that the function won't tell you if it's exotic in your area, a lot of exotic species are not considered invaders (yet).

As expected, the function is as good as the database is, which I find quite reliable and well maintained. The database is also able to recognize a lot (but not all) of the species synonyms.

Note that `eol` with source of gisd or gisd100 may end up with different results as this function goes directly to the GISD website, whereas EOL only updates their GISD data occassionally. See notes in `eol`.

Usage

```
gisd(x, simplify = FALSE, messages = TRUE, ...)
```

Arguments

`x`	character; a vector of scientific species names in the form of c("Genus species").
`simplify`	logical; returns a data.frame with the species name and the values "Invasive", "Not in GISD". I recomend to check first the not simplified version (default), which contains raw information about the level of invasiveness.
`messages`	logical; If `TRUE` (default), informative messages printed.
`...`	curl options passed on to `HttpClient`

Value

A list with species names, native range countries, and invasive range countries

Author(s)

Ignasi Bartomeus <nacho.bartomeus@gmail.com>

Examples

```
## Not run:
sp <- c("Carpobrotus edulis", "Rosmarinus officinalis")
## first species is invasive, second one is not.
gisd(sp)
gisd(sp, simplify = TRUE)
```

```
sp <- c("Carpobrotus edulis", "Rosmarinus officinalis", "Acacia mangium",
"Archontophoenix cunninghamiana", "Antigonon leptopus")
gisd(sp)
gisd(sp, simplify = TRUE)

## End(Not run)
```

`griis` *Check invasive species status for a species from GRIIS database*

Description

This retrieves information from GRIIS (http://www.griis.org/) and returns all the queried records. As other functions in this package, the function is as good as the database is.

Usage

```
griis(name = NULL, impacts = NULL, verified = NULL, country = NULL,
  kindom = NULL, type = NULL, ...)
```

Arguments

`name`	character; a string with the scientific species name in the form of "Genus species". Default is NULL: return all records.
`impacts`	character; "Yes" for returning only records with impacts. Default to NULL: return all records.
`verified`	character; "Yes" for returning only verified records. Default to NULL: return all records.
`country`	character containing a valid name of a country for which to filter the results. Default to NULL: return all records.
`kindom`	character containing a valid name of a kindom (plantae, animalia, fungi, protozoa, chromista, others,) for which to filter the results. Default to NULL: return all records.
`type`	character containing a valid name of a environment type (terrestrial, freshwater, marine, brackish, host) for which to filter the results. Default to NULL: return all records.
`...`	curl options passed on to `HttpClient`

Value

A data.frame with species names, country where recorded, origin and source among other fields.

Note

It seems as 'name' overrides 'kindom', which means records from a a plant species will be returned even if kindom is set to animalia.

Author(s)

Ignasi Bartomeus <nacho.bartomeus@gmail.com>

Examples

```
## Not run:
griis(name = "Carpobrotus edulis")
griis(name = "Carpobrotus edulis", country = "Portugal")

## End(Not run)
```

`is_native` *Check if a species is native somewhere*

Description

This function check the status (native or exotic) of a species in a given place

For that end, calls `itis_native` and `flora_europaea`. See help documentation of those functions for details.

So many more things can be done, like checking species first with **taxize**, adding more native lists to check...

Usage

```
is_native(sp, where, region = c("america", "europe"), ...)
```

Arguments

`sp`	character; a vector of length one with a single scientific species names in the form of `c("Genus species")`.
`where`	character; a vector of length one with a single place. For America has to match one of those: "Continental US", "Alaska", "Canada", "Caribbean Territories", "Central Pacific Territories", "Hawaii", "Mexico". For Europe has to match one of those: "Albania", "Austria", "Azores", "Belgium", "Islas_Baleares", "Britain", "Bulgaria", "Corse", "Kriti", "Czechoslovakia", "Denmark", "Faroer", "Finland", "France", "Germany", "Greece", "Ireland", "Switzerland", "Netherlands", "Spain", "Hungary", "Iceland", "Italy", "Jugoslavia", "Portugal", "Norway", "Poland", "Romania", "USSR", "Sardegna", "Svalbard", "Sicilia", "Sweden", "Turkey", "USSR_Northern_Division", "USSR_Baltic_Division", "USSR_Central_Division", "USSR_South_western", "USSR_Krym", "USSRSouth_eastern_Division"
`region`	character; a vector of length one with a single region. Only "europe" and "america" implemented "europe" checks Flora Europaea and only contain plants. "america" checks ITIS and contain both plant and animals.
`...`	Curl options passed on to `HttpClient`

Value

A data.frame, with species name and result of origin check

Author(s)

Ignasi Bartomeus <nacho.bartomeus@gmail.com>

Examples

```
## Not run:
sp <- c("Lavandula stoechas", "Carpobrotus edulis", "Rhododendron ponticum",
       "Alkanna lutea", "Anchusa arvensis")
is_native(sp[1], where = "Islas_Baleares", region = "europe")
lapply(sp, is_native, where = "Continental US", region = "america")
lapply(sp, is_native, where = "Islas_Baleares", region = "europe")

# combine output for many taxa
res <- lapply(sp, is_native, where = "Continental US", region = "america")
do.call(rbind, res)

## End(Not run)
```

---

nsr *Search the Native Species Resolver*

Description

Search the Native Species Resolver

Usage

```
nsr(species, country, stateprovince = NULL, countyparish = NULL, ...)
```

Arguments

species	(character) One or more species names. required.
country	(character) A country name. required.
stateprovince	(character) A state or province name
countyparish	(character) A county or parish name
...	curl options passed on to crul::HttpClient

Details

Currently, only one name is allowed per request. We loop internally over a list of length > 1, but this will still be slow due to only 1 name per request.

Note that this service can be quite slow.

political names

- nsr_countries: is a vector of country names that we use to check your country names
- nsr_pol_divisions: is a data.frame of country names and state/province names that we used to check your parameter inputs - these are for checklists that NSR has complete coverage for

References

http://bien.nceas.ucsb.edu/bien/tools/nsr/nsr-ws/

Examples

```
## Not run:
nsr("Pinus ponderosa", country = "United States")
nsr(c("Pinus ponderosa", "Poa annua"), country = "United States")
splist <- c("Pinus ponderosa", "Poa annua", "bromus tectorum", "Ailanthus altissima")
nsr(splist, country = "United States")
nsr(splist, country = "United States", stateprovince = "California")

# curl options
nsr("Pinus ponderosa", "United States", verbose = TRUE)

## End(Not run)
```

---

nsr_countries *Vector of country names for use with NSR*

Description

Vector of country names for use with NSR

Format

A vector of countries of length 251

---

nsr_pol_divisions *NSR political divisions*

Description

NSR political divisions

Format

A data frame with 73 rows and 2 variables:

country Country name

state_province State or province name

Index