Package 'rbison'

April 11, 2018

Title Interface to the 'USGS' 'BISON' 'API'

Description Interface to the 'USGS' 'BISON' (<https://bison.usgs.gov/>) 'API', a 'database' for species occurrence data. Data comes from species in the United States from participating data providers. You can get data via 'taxonomic' and location based queries. A simple function is provided to help visualize data.

Version 0.6.0

License MIT + file LICENSE

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

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

LazyData true

VignetteBuilder knitr

Imports plyr, crul (>= 0.3.4), ggplot2, mapproj, grid, sp, dplyr (>= 0.5.0), jsonlite (>= 1.1)

Suggests roxygen2 (>= 6.0.1), knitr, testthat, taxize

RoxygenNote 6.0.1

X-schema.org-applicationCategory Data Access

X-schema.org-keywords species, occurrences, biodiversity, maps

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

NeedsCompilation no

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

Maintainer Scott Chamberlain <myrmecocystus@gmail.com>

Repository CRAN

Date/Publication 2018-04-11 05:48:01 UTC

R topics documented:

`rbison-package` *rbison is an interface to the USGS Bison API.*

Description

To get started, see the vignette `vignette(package="rbison")`

Details

See https://bison.usgs.gov/doc/api.jsp for API docs for the BISON API.

To cite rbison, do `citation(package='rbison')`

Use the following format to cite data retrieved from BISON:

Biodiversity occurrence data published by: (Accessed through Biodiversity Information Serving our Nation (BISON), bison.usgs.gov, YYYY-MM-DD).

For example:

Biodiversity occurrence data published by: Field Museum of Natural History, Museum of Vertebrate Zoology, University of Washington Burke Museum, and University of Turku (Accessed through Biodiversity Information Serving our Nation (BISON), bison.usgs.gov, 2013-04-22).

Base URL for the BISON API: https://bison.usgs.gov

Author(s)

Scott Chamberlain <myrmecocystus@gmail.com>

`all_states` *Data for a states map*

Description

Data for a states map

bison *Search for and collect data from the USGS Bison API.*

Description

Search for and collect data from the USGS Bison API.

Usage

```
bison(species = NULL, type = "scientific_name", tsn = NULL, start = 0,
  count = 25, countyFips = NULL, county = NULL, state = NULL,
  aoi = NULL, aoibbox = NULL, params = NULL, what = "all", ...)
```

Arguments

species	(character) A species name.
type	(character) Type, one of scientific_name or common_name.
tsn	(numeric) Specifies the TSN to query by. If you supply a tsn it doesn't make sense to supply a species name as well. Example:162003.
start	(numeric) Record to start at. Default: 0
count	(numeric) Number of records to return. Default: 0. Max: 500
countyFips	(character) Specifies the county fips code to geographically constrain the search to one county. Character must be supplied as a number starting with zero may lose the zero. Eg: "49015".
county	(character) County name. As codes are a pain in the ass, you can put in the county name here instead of specifying a countyFips entry, and bison will attempt to look up the countyFips code. (character)
state	(character) Specifies the state name to geographically constrain the search. Example: Tennessee.
aoi	(character) Specifies a WKT (Well-Known Text) polygon to geographically constrain the search.Eg.: c(-111.06 38.84, -110.80 39.377, -110.20 39.17, -110.20 38.90, -110.63 38.67, -111.06 38.84), which calls up the occurrences within the specified area. Check out the Wikipedia page here http://en.wikipedia.org/wiki/Well-known_text for an in depth look at the options, terminology, etc.
aoibbox	(character) Specifies a four-sided bounding box to geographically constrain the search (using format: minx,miny,maxx,maxy). The coordinates are Spherical Mercator with a datum of WGS84. Example: -111.31,38.81,-110.57,39.21
params	(character) String of parameters, one of providerID, resourceID, basisOfRecord, catalogNumber, year, computedStateFips, hierarchy_homonym_string, TSNs, recordedBy, occurrenceID, collectorNumber, provider, ownerInstitutionCollectionCode, eventDate, providedScientificName, scientificName, ITISscientificName, providedCommonName, ITIScommonName, kingdom, ITIStsn, centroid, higherGeographyID, computedCountyFips, providedCounty, calculatedCounty, stateProvince, calculatedState, countryCode. See examples.

what	What to return? One of 'all', 'summary', 'points', 'counties', 'states', 'raw', or 'list'. All data is returned from the BISON API, but this parameter lets you select just the parts you want, and the rest is discarded before returning the result to you.
...	Further args passed on to crul::HttpClient(). See examples.

References

https://bison.usgs.gov/#opensearch

See Also

bison_solr() bison_tax()

Examples

```
## Not run:
bison(species="Bison bison", count=50, what='summary')
bison(species="Bison bison", count=50, what='points')
bison(species="Bison bison", count=50, what='counties')
bison(species="Bison bison", count=50, what='states')
bison(species="Bison bison", count=50, what='raw')
bison(species="Bison bison", count=50, what='list')

out <- bison(species="Bison bison", count=50) # by default gets 10 results
out$summary # see summary
out$counties # see county data
out$states # see state data
bisonmap(out, tomap = "points")
bisonmap(out, tomap = "county")
bisonmap(out, tomap = "state")

# Search for a common name
bison(species="Tufted Titmouse", type="common_name", what='summary')

# Constrain search to a certain county, 49015 is Emery County in Utah
bison(species="Helianthus annuus", countyFips = "49015")

# Constrain search to a certain county, specifying county name instead of
# code
bison(species="Helianthus annuus", county = "Los Angeles")
bison(species="Helianthus annuus", county = "Los")

# Constrain search to a certain aoi, which turns out to be Emery County,
# Utah as well
bison(species="Helianthus annuus",
 aoi = "POLYGON((-111.06360117772908 38.84001566645886,
                 -110.80542246679359 39.37707771107983,
                 -110.20117441992392 39.17722368276862,
                 -110.20666758398464 38.90844075244811,
                 -110.63513438085685 38.67724220095734,
                 -111.06360117772908 38.84001566645886))")
```

```
# Constrain search to a certain aoibbox, which, you guessed it, is also
# Emery Co., Utah
bison(species="Helianthus annuus", aoibbox = '-111.31,38.81,-110.57,39.21')

# Taxonomic serial number
bison(tsn = 162003)
## If you don't have tsn's, search for a taxonomic serial number
library('taxize')
poa_tsn <- get_tsn('Poa annua')
bison(tsn = poa_tsn)

# Curl debugging and some of these examples aren't
# that useful, but are given for demonstration purposes
## get curl verbose output to see what's going on with your request
bison(tsn = 162003, count=1, what="points", verbose = TRUE)
## set a timeout so that the call stops after time x, compare 1st to 2nd call
# bison(tsn=162003, count=1, what="points", timeout_ms = 1)
## set cookies
bison(tsn=162003, count=1, what="points", cookie = "a=1;b=2")
## user agent and verbose
bison(tsn=162003, count=1, what="points", useragent = "rbison",
  verbose = TRUE)

# Params - the params function accepts a number of search terms
## Find the provider with ID 318.
bison(params='providerID:("318")')
## Find all resources with id of '318,1902' OR '318,9151', with values
## separated by spaces.
bison(params='resourceID:("318,1902" "318,9151")')
## Criterion may be combined using the semicolon (';') character, which
## translates to a logical AND operator. Note that field names and values
## are case sensitive.
bison(params='providerID:("408" "432");resourceID:("14027")')
## Search by basisOfRecord, for specimen types in this case
bison(params='basisOfRecord:(specimen)')
## Search by computedStateFips, 01 for Alabama
bison(params='computedStateFips:01')
## Search by ITIStsn
bison(params='ITIStsn:162003')
## Search by countryCode
bison(params='countryCode:US')
## Search by ITIScommonName
bison(params='ITIScommonName:"Canada goose"')

## End(Not run)
```

`bisonmap` *Make map to visualize BISON data.*

Description

Make map to visualize BISON data.

Usage

```
bisonmap(input = NULL, tomap = "points", geom = geom_point,
  jitter = NULL, customize = NULL)

## S3 method for class 'bison'
bisonmap(input = NULL, tomap = "points",
  geom = geom_point, jitter = NULL, customize = NULL)

## S3 method for class 'bison_solr'
bisonmap(input = NULL, tomap = "points",
  geom = geom_point, jitter = NULL, customize = NULL)
```

Arguments

input	Input bison object.
tomap	One of points (occurrences), county (counts by county), or state (counts by state).
geom	geom_point or geom_jitter, not quoted.
jitter	jitter position, see ggplot2 help.
customize	Pass in more to the plot.

Value

Map (using ggplot2 package) of points on a map.

Examples

```
## Not run:
# Using function bison
library("ggplot2")
out <- bison(species="Accipiter", type="scientific_name", count=300)
bisonmap(input=out)
bisonmap(input=out, geom=geom_jitter, jitter=position_jitter(width = 0.3,
  height = 0.3))

# Using function bison_solr
out <- bison_solr(scientificName='Ursus americanus', rows=200)
bisonmap(out)

## End(Not run)
```

`bison_datause` *Get BISON data use agreement details and examples for how to cite data.*

Description

Get BISON data use agreement details and examples for how to cite data.

Usage

```
bison_datause()

bison_citation()
```

References

https://bison.usgs.gov/doc/api.jsp#data

`bison_providers` *Get information about BISON data providers.*

Description

Get information about BISON data providers.

Usage

```
bison_providers(details = FALSE, provider_no = NULL, ...)
```

Arguments

`details`	(logical) If `TRUE`, returns a list of data.frame's for each provider, including their resource details. If `FALSE` (default), only coarse grained data returned.
`provider_no`	(numeric) Provider number. If this parameter is provided, details is forced to be `FALSE`
`...`	Further args passed on to `crul::HttpClient()` See examples in `bison()`

Value

A data.frame or list of data.frame's

Examples

```
## Not run:
head(bison_providers())
head(bison_providers(provider_no=131))
out <- bison_providers(details=TRUE)
out$National_Herbarium_of_New_South_Wales

## End(Not run)
```

`bison_solr` *Search for and collect occurrence data from the USGS Bison API using their solr endpoint.*

Description

This fxn is somewhat similar to `bison()`, but interacts with the SOLR interface https://bison.usgs.gov/#solr instead of the OpenSearch interface https://bison.usgs.gov/#opensearch, which `bison()` uses.

Usage

```
bison_solr(decimalLatitude = NULL, decimalLongitude = NULL, year = NULL,
  providerID = NULL, resourceID = NULL, pointPath = NULL,
  basisOfRecord = NULL, eventDate = NULL, computedCountyFips = NULL,
  computedStateFips = NULL, scientificName = NULL,
  hierarchy_homonym_string = NULL, TSNs = NULL, recordedBy = NULL,
  occurrenceID = NULL, catalogNumber = NULL, ITIScommonName = NULL,
  kingdom = NULL, collectorNumber = NULL, provider = NULL,
  ownerInstitutionCollectionCode = NULL, providedScientificName = NULL,
  ITISscientificName = NULL, providedCommonName = NULL, ITIStsn = NULL,
  centroid = NULL, higherGeographyID = NULL, providedCounty = NULL,
  calculatedCounty = NULL, stateProvince = NULL, calculatedState = NULL,
  countryCode = NULL, callopts = list(), verbose = TRUE, ...)
```

Arguments

`decimalLatitude`	Geographic coordinate that specifies the north south position of a location on the Earth surface.
`decimalLongitude`	Geographic coordinate that specifies the east-west position of a location on the Earth surface.
`year`	The year the collection was taken.
`providerID`	(character) Unique identifier assigned by GBIF.
`resourceID`	(character) A unique identifier that is a concatentation of the provider identifier and the resource id seperated by a comma.

`pointPath` A dynamic field that contains the location in longitude and latitude followed by the basis of record and an optional Geo (Spatial) precision. Geo (Spatial) precision is an added descriptor when the record is a county centroid.

`basisOfRecord` One of these enumerated values: Observation, Germplasm, Fossil, Specimen, Literature, Unknown, or Living.

`eventDate` The date when the occurrence was recorded. Dates need to be of the form YYYY-MM-DD

`computedCountyFips`
County FIPS code conforming to standard FIPS 6-4 but with leading zeros removed. See fips dataset for codes

`computedStateFips`
The normalized state fips code. See fips dataset for codes

`scientificName` The species scientific name that is searchable in a case insensitive way.

`hierarchy_homonym_string`
hierarachy of the accepted or valid species name starting at kingdom. If the name is a taxonomic homonym more than one string is provided seperated by ';'.

`TSNs` Accepted or valid name is provided. If the name is a taxonmic homonym more than one TSN is provided.

`recordedBy` Individual responsible for the scientific record.

`occurrenceID` Non-persistent unique identifier.

`catalogNumber` Unique key for every record (occurrence/row) within a dataset that is not manipulated nor changed (nor generated, if not provided) during the data ingest.

`ITIScommonName` Common name(s) from ITIS, e.g. "Canada goose"

`kingdom` Kingdom name, from GBIF raw occurrence or BISON provider.

`collectorNumber`
An identifier given to the occurrence at the time it was recorded, such as a specimen collector's number. / e.g., "SJM030022".

`provider` Non-persistent unique identifier.

`ownerInstitutionCollectionCode`
Name for the dataset, format = OwnerInstitution-Collection. / e.g., "USGS NAWQA BioData - Fish Occurrence Records"

`providedScientificName`
Full scientific name as provided in the dataset, with authorship and date information if known.

`ITISscientificName`
Scientific name from join on ITIS table, calculated e.g, "Bison bison"

`providedCommonName`
A list (concatenated and separated) of the available vernacular species names. / e.g., "common shrew, Masked Shrew"

`ITIStsn` Phase II: ITIS TSN corresponding to clean_provided_scientific_name. May be invalid,unaccepted. Calculated. e.g., "3250", "05713"

`centroid` Text string indicating that provided lat/lon point represents a polygon centroid. Text provides description of the centroid.

`higherGeographyID`
: 5-digit numeric text string geographic code for the state-county combination provided by data provider. / e.g,. "13029"

`providedCounty`
: Full county, parish, or organized borough name, as provided in the dataset. If provided, Verbatum State is required. Is not changed during data ingest. / e.g., "Fairfax"

`calculatedCounty`
: Full county, parish, or organized borough name of the occurrence calculated. / e.g., "Fairfax"

`stateProvince`
: Full name of state or territory of the occurrence, as provided in the dataset.

`calculatedState`
: U.S. State or territory name calculated. e.g., "Puerto Rico"

`countryCode`
: The geographic location of the specific occurrence, expressed through a constrained vocabulary of countries using 2-letter ISO country code.

`callopts`
: Further args passed on to `crul::HttpClient()` for HTTP debugging/inspecting. In `bison`, `bison_providers`, and `bison_stats`, `...` is used instead of callopts, but `...` is used here to pass additional Solr params.

`verbose`
: Print message with url (`TRUE`, default).

`...`
: Additional SOLR query arguments. See details.

Details

Named parameters in this function are combined with `AND` and passed on to `q` SOLR parameter. Of course parameters can be more flexibly combined - let us know if you want that flexibility and we can think about that.

Value

An object of class bison_solr - which is a list with slots for number of records found (num_found), records, highlight, or facets.

SOLR search parameters passed on via ...

- fl: Fields to return in the query
- rows: Number of records to return
- sort: Field to sort by, see examples
- facet: Facet or not, logical
- facet.field: Fields to facet by

You can also use highlighting in solr search, but I'm not sure I see a use case for it with BISON data, though you can do it with this function.

For a tutorial see here http://lucene.apache.org/solr/3_6_2/doc-files/tutorial.html

Range searches

If you pass a vector of length 2 to a parameter we construct a range query for you. For example, `c(4, 5)` turns into `[4 TO 5]`. The `[]` syntax means the search is inclusive, meaning 4 to 5, including 4 and 5. Let us know if you think you need more flexible searching. That is, doing exclusive `{}` or mixed searches (`{]` or `[}`). Range searches can only be done with variables that are numeric/integer or dates or strings that can be coerced to dates. Dates need to be of the form YYYY-MM-DD

References

https://bison.usgs.gov/#solr

See Also

`bison_tax()`, `bison()`

The USGS BISON Solr installation version as of 2014-10-14 was 4.4.

Examples

```
## Not run:
x=bison_solr(scientificName='Ursus americanus')

bison_solr(scientificName='Ursus americanus', computedStateFips='02',
 fl="scientificName")

x <- bison_solr(scientificName='Ursus americanus', computedStateFips='02', rows=50)
x$points$computedStateFips
head(x$points)

bison_solr(ITISscientificName='Ursus americanus', rows=50)

bison_solr(providerID = 220)

# combining parameters
x <- bison_solr(eventDate = c('2008-01-01', '2010-12-31'),
  ITISscientificName="Helianthus annuus", rows = 100)
head(x$points)
sort(x$points$eventDate)

# range queries
## range search with providerID
bison_solr(providerID = c(220, 221))
## date range search
x <- bison_solr(eventDate = c('2010-08-08', '2010-08-21'))
sort(x$points$eventDate)
## TSN range search
x <- bison_solr(TSNs = c(174773, 174775), rows = 100)
sort(x$points$TSN)
## can't do range searches with character strings (that are not dates)
# bison_solr(kingdom = c("Animalia", "Plantae"))

# more examples
```

```
bison_solr(TSNs = 174773)
bison_solr(occurrenceID = 576630651)
bison_solr(catalogNumber = 'OBS101299944')
bison_solr(ITIScommonName = "Canada goose")
bison_solr(kingdom = "Animalia")
bison_solr(kingdom = "Plantae")

# Mapping
out <- bison_solr(scientificName='Ursus americanus', rows=200)
bisonmap(out)

out <- bison_solr(scientificName='Helianthus annuus', rows=800)
bisonmap(out)

# Using additional solr fields
## Faceting
bison_solr(scientificName='Helianthus annuus', rows=0, facet='true',
 facet.field='computedStateFips')

## Highlighting
bison_solr(scientificName='Helianthus annuus', rows=10, hl='true',
 hl.fl='scientificName')

## Use of hierarchy_homonym_string
bison_solr(hierarchy_homonym_string = '-202423-914154-914156-158852-')
## -- This is a bit unwieldy, but you can find this string in the output
## of a call, like this
x <- bison_solr(scientificName='Ursus americanus', rows=1)
string <- x$points$hierarchy_homonym_string
bison_solr(hierarchy_homonym_string = string)

# The pointPath parameter
bison_solr(pointPath = '/-110.0,45.0/specimen')

# Curl options
bison_solr(scientificName='Ursus americanus', callopts=list(verbose = TRUE))

## End(Not run)
```

`bison_stats` *Get statistics about BISON downloads.*

Description

Get statistics about BISON downloads.

Usage

```
bison_stats(what = "stats", ...)
```

Arguments

what	(character) One of stats (default), search, downnload, or wms. See Details.
...	Further args passed on to crul::HttpClient(). See examples in bison

Details

For the 'what' parameter:

- stats - Retrieve all data provider accumulated statistics.
- search - Retrieve data provider statistics for BISON searches.
- download - Retrieve data provider statistics for data downloads from BISON.
- wms - Retrieve data provider statistics for BISON OGC WMS tile requests.

Value

A list of data frame's with names of the list the different data sources

Examples

```
## Not run:
out <- bison_stats()
out <- bison_stats(what='search')
out <- bison_stats(what='download')
out <- bison_stats(what='wms')
out$Arctos
out$Harvard_University_Herbaria
out$ZooKeys

## End(Not run)
```

---

bison_tax *Search for and collect taxonomic name data from the USGS Bison API using solr*

Description

See the SOLR documentation here http://lucene.apache.org/solr/ for other parameters you can use.

The following two methods are possible, as far as I know you can only use one at a time:

- vernacularName The species specific common names that is searchable in a case insensitive way.
- scientificName The species scientific name that is associated with a common name that is searchable in a case insensitive way.

Usage

```
bison_tax(query, method = "vernacularName", exact = FALSE, parsed = TRUE,
  callopts = list(), ...)
```

Arguments

query	Name to search for. Required.
method	The field to query by. See description below for details.
exact	Exact matching or not. See examples. Defaults to FALSE.
parsed	If TRUE (default) creates data.frame of names data output. Otherwise, a list.
callopts	Further args passed on to crul::HttpClient() for HTTP debugging/inspecting. In bison, bison_providers, and bison_stats, ... is used instead of callopts, but ... is used here to pass additional Solr params.
...	Further solr arguments passed in to the query. See examples below.

Value

A list.

See Also

bison_solr(), bison()

Examples

```
## Not run:
# All taxa
bison_tax("*:*")

# Some example calls
bison_tax(query="*bear")
bison_tax(query="Helianthus", method="scientificName")

# Exact argument, here nothing found with latter call as '*bear'
# doesn't exist, which makes sense
bison_tax(query="*bear", exact=FALSE)
bison_tax(query="*bear", exact=TRUE)

# Using solr arguments (not all Solr arguments work)
## Return a certain number of rows
bison_tax(query="*bear", method="vernacularName", rows=3)
## Return certain fields
bison_tax(query="*bear", method="vernacularName", fl='vernacularName')

# Curl options
bison_tax(query='*dolphin', callopts=list(verbose = TRUE))

## End(Not run)
```

fips *Fips codes for states and counties*

Description

See https://www.census.gov/geo/reference/codes/cou.html and https://www.census.gov/geo/reference/ansi_statetables.html for more information on FIPS codes

Format

A data frame with 3142 rows and 4 variables:

state State name

county County name

fips_state State FIPS code

fips_county County FIPS code

is.bison *Check if object is of class bison*

Description

Check if object is of class bison

Usage

```
is.bison(x)
```

Arguments

x	input

is.bison_solr *Check if object is of class bison_solr*

Description

Check if object is of class bison_solr

Usage

```
is.bison_solr(x)
```

Arguments

x	input

Index