Package 'bold'

June 18, 2020

Title Interface to Bold Systems API

Description A programmatic interface to the Web Service methods provided by Bold Systems (<http://www.boldsystems.org/>) for genetic 'barcode' data. Functions include methods for searching by sequences by taxonomic names, ids, collectors, and institutions; as well as a function for searching for specimens, and downloading trace files.

Version 1.1.0

License MIT + file LICENSE

URL https://docs.ropensci.org/bold, https://github.com/ropensci/bold

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

VignetteBuilder knitr

LazyData yes

Encoding UTF-8

Imports xml2, crul (>= 0.3.8), stringr, jsonlite, reshape, plyr, data.table, tibble

Suggests sangerseqR, knitr, testthat, vcr (>= 0.5.4)

RoxygenNote 7.1.0

X-schema.org-applicationCategory Data Access

X-schema.org-keywords biodiversity, barcode, DNA, sequences, fasta

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 2020-06-17 23:30:07 UTC

R topics documented:

`bold-package` *bold*

Description

bold: A programmatic interface to the Barcode of Life data

About

This package gives you access to data from BOLD System http://www.boldsystems.org/ via their API (http://v4.boldsystems.org/index.php/api_home)

Functions

- `bold_specimens()` - Search for specimen data
- `bold_seq()` - Search for and retrieve sequences
- `bold_seqspec()` - Get sequence and specimen data together
- `bold_trace()` - Get trace files - saves to disk
- `read_trace()` - Read trace files into R
- `bold_tax_name()` - Get taxonomic names via input names
- `bold_tax_id()` - Get taxonomic names via BOLD identifiers
- `bold_identify()` - Search for match given a COI sequence

Interestingly, they provide xml and tsv format data for the specimen data, while they provide fasta data format for the sequence data. So for the specimen data you can get back raw XML, or a data frame parsed from the tsv data, while for sequence data you get back a list (b/c sequences are quite long and would make a data frame unwieldy).

`bold_filter` *Filter BOLD specimen + sequence data (output of bold_seqspec)*

Description

Picks either shortest or longest sequences, for a given grouping variable (e.g., species name)

Usage

```
bold_filter(x, by, how = "max")
```

Arguments

`x`	(data.frame) a data.frame, as returned from `bold_seqspec()`. Note that some combinations of parameters in `bold_seqspec()` don't return a data.frame. Stops with error message if this is not a data.frame. Required.
`by`	(character) the column by which to group. For example, if you want the longest sequence for each unique species name, then pass **species_name**. If the column doesn't exist, error with message saying so. Required.
`how`	(character) one of "max" or "min", which get used as `which.max` or `which.min` to get the longest or shortest sequence, respectively. Note that we remove gap/alignment characters (`-`)

Value

a tibble/data.frame

Examples

```
## Not run:
res <- bold_seqspec(taxon='Osmia')
maxx <- bold_filter(res, by = "species_name")
minn <- bold_filter(res, by = "species_name", how = "min")

vapply(maxx$nucleotides, nchar, 1, USE.NAMES = FALSE)
vapply(minn$nucleotides, nchar, 1, USE.NAMES = FALSE)

## End(Not run)
```

`bold_identify` *Search for matches to sequences against the BOLD COI database.*

Description

Search for matches to sequences against the BOLD COI database.

Usage

```
bold_identify(sequences, db = "COX1", response = FALSE, ...)
```

Arguments

`sequences`	(character) Returns all records containing matching marker codes. Required. One or more. See Details.
`db`	(character) The database to match against, one of COX1, COX1_SPECIES, COX1_SPECIES_PUBLIC, OR COX1_L604bp. See Details for more information.
`response`	(logical) Note that response is the object that returns from the Curl call, useful for debugging, and getting detailed info on the API call.
`...`	Further args passed on to crul::verb-GET, main purpose being curl debugging BOLD only allows one sequence per query. We internally `lapply` over the input values given to the sequences' parameter to search with one sequence per query. Remember this if you have a lot of sequences - you are doing a separate query for each one, so it can take a long time - if you run into errors let us know.

Value

A data.frame with details for each specimen matched. if a failed request, returns `NULL`

db parmeter options

- COX1 Every COI barcode record on BOLD with a minimum sequence length of 500bp (warning: unvalidated library and includes records without species level identification). This includes many species represented by only one or two specimens as well as all species with interim taxonomy. This search only returns a list of the nearest matches and does not provide a probability of placement to a taxon.
- COX1_SPECIES Every COI barcode record with a species level identification and a minimum sequence length of 500bp. This includes many species represented by only one or two specimens as well as all species with interim taxonomy.
- COX1_SPECIES_PUBLIC All published COI records from BOLD and GenBank with a minimum sequence length of 500bp. This library is a collection of records from the published projects section of BOLD.
- OR COX1_L604bp Subset of the Species library with a minimum sequence length of 640bp and containing both public and private records. This library is intended for short sequence identification as it provides maximum overlap with short reads from the barcode region of COI.

Named outputs

To maintain names on the output list of data make sure to pass in a named list to the sequences parameter. You can for example, take a list of sequences, and use `stats::setNames()` to set names.

References

http://v4.boldsystems.org/index.php/resources/api?type=idengine

See Also

`bold_identify_parents()`

Examples

```
## Not run:
seq <- sequences$seq1
res <- bold_identify(sequences=seq)
head(res[[1]])
head(bold_identify(sequences=seq, db='COX1_SPECIES')[[1]])

## End(Not run)
```

---

bold_identify_parents *Add taxonomic parent names to a data.frame*

Description

Add taxonomic parent names to a data.frame

Usage

```
bold_identify_parents(
  x,
  wide = FALSE,
  taxid = NULL,
  taxon = NULL,
  tax_rank = NULL,
  tax_division = NULL,
  parentid = NULL,
  parentname = NULL,
  taxonrep = NULL,
  specimenrecords = NULL,
  ...
)
```

Arguments

x	(data.frame/list) list of data.frames - the output from a call to bold_identify(). or a single data.frame from the output from same. required.
wide	(logical) output in long or wide format. See Details. Default: FALSE
taxid	(character) A taxid name. Optional. See Filtering below.
taxon	(character) A taxon name. Optional. See Filtering below.
tax_rank	(character) A tax_rank name. Optional. See Filtering below.
tax_division	(character) A tax_division name. Optional. See Filtering below.
parentid	(character) A parentid name. Optional. See Filtering below.
parentname	(character) A parentname name. Optional. See Filtering below.
taxonrep	(character) A taxonrep name. Optional. See Filtering below.
specimenrecords	(character) A specimenrecords name. Optional. See Filtering below.
...	Further args passed on to crul::verb-GET, main purpose being curl debugging

Details

This function gets unique set of taxonomic names from the input data.frame, then queries bold_tax_name() to get the taxonomic ID, passing it to bold_tax_id() to get the parent names, then attaches those to the input data.

Records in the input data that do not have matches for parent names simply get NA values in the added columns.

Value

a list of the same length as the input

Filtering

The parameters taxid, taxon, tax_rank, tax_division, parentid, parentname,taxonrep, and specimenrecords are not used in the search sent to BOLD, but are used in filtering the data down to a subset that is closer to the target you want. For all these parameters, you can use regex strings since we use grep() internally to match. Filtering narrows down to the set that matches your query, and removes the rest. The data.frame that we filter on with these parameters internally is the result of a call to the bold_tax_name() function.

wide vs long format

When wide = FALSE you get many rows for each record. Essentially, we cbind the taxonomic classification onto the one row from the result of bold_identify(), giving as many rows as there are taxa in the taxonomic classification.

When wide = TRUE you get one row for each record - thus the dimensions of the input data stay the same. For this option, we take just the rows for taxonomic ID and name for each taxon in the taxonomic classification, and name the columns by the taxon rank, so you get phylum and phylum_id, and so on.

Examples

```
## Not run:
df <- bold_identify(sequences = sequences$seq2)

# long format
out <- bold_identify_parents(df)
str(out)
head(out[[1]])

# wide format
out <- bold_identify_parents(df, wide = TRUE)
str(out)
head(out[[1]])

x <- bold_seq(taxon = "Satyrium")
out <- bold_identify(c(x[[1]]$sequence, x[[13]]$sequence))
res <- bold_identify_parents(out)
res

x <- bold_seq(taxon = 'Diplura')
out <- bold_identify(vapply(x, "[[", "", "sequence")[1:20])
res <- bold_identify_parents(out)

## End(Not run)
```

`bold_seq` *Search BOLD for sequences.*

Description

Get sequences for a taxonomic name, id, bin, container, institution, researcher, geographic, place, or gene.

Usage

```
bold_seq(
  taxon = NULL,
  ids = NULL,
  bin = NULL,
  container = NULL,
  institutions = NULL,
  researchers = NULL,
  geo = NULL,
  marker = NULL,
  response = FALSE,
  ...
)
```

Arguments

taxon	(character) Returns all records containing matching taxa. Taxa includes the ranks of phylum, class, order, family, subfamily, genus, and species.
ids	(character) Returns all records containing matching IDs. IDs include Sample IDs, Process IDs, Museum IDs and Field IDs.
bin	(character) Returns all records contained in matching BINs. A BIN is defined by a Barcode Index Number URI.
container	(character) Returns all records contained in matching projects or datasets. Containers include project codes and dataset codes
institutions	(character) Returns all records stored in matching institutions. Institutions are the Specimen Storing Site.
researchers	(character) Returns all records containing matching researcher names. Researchers include collectors and specimen identifiers.
geo	(character) Returns all records collected in matching geographic sites. Geographic sites includes countries and province/states.
marker	(character) Returns all records containing matching marker codes.
response	(logical) Note that response is the object that returns from the Curl call, useful for debugging, and getting detailed info on the API call.
...	Further args passed on to `crul::verb-GET`, main purpose being curl debugging

Value

A list with each element of length 4 with slots for id, name, gene, and sequence.

Large requests

Some requests can lead to errors. These often have to do with requesting data for a rank that is quite high in the tree, such as an Order, for example, Coleoptera. If your request is taking a long time, it's likely that something will go wrong on the BOLD server side, or we'll not be able to parse the result here in R because R can only process strings of a certain length. `bold` users have reported errors in which the resulting response from BOLD is so large that we could not parse it.

A good strategy for when you want data for a high rank is to do many separate requests for lower ranks within your target rank. You can do this manually, or use the function `taxize::downstream` to get all the names of a lower rank within a target rank. There's an example in the README (https://docs.ropensci.org/bold/#large-data)

If a request times out

This is likely because you're request was for a large number of sequences and the BOLD service timed out. You still should get some output, those sequences that were retrieved before the time out happened. As above, see the README (https://docs.ropensci.org/bold/#large-data) for an example of dealing with large data problems with this function.

Marker

Notes from BOLD on the `marker` param: "All markers for a specimen matching the search string will be returned. ie. A record with COI-5P and ITS will return sequence data for both markers even if only COI-5P was specified."

You will likely end up with data with markers that you did not request - just be sure to filter those out as needed.

References

http://v4.boldsystems.org/index.php/resources/api?type=webservices

Examples

```
## Not run:
res <- bold_seq(taxon='Coelioxys')
bold_seq(taxon='Aglae')
bold_seq(taxon=c('Coelioxys','Osmia'))
bold_seq(ids='ACRJP618-11')
bold_seq(ids=c('ACRJP618-11','ACRJP619-11'))
bold_seq(bin='BOLD:AAA5125')
bold_seq(container='ACRJP')
bold_seq(researchers='Thibaud Decaens')
bold_seq(geo='Ireland')
bold_seq(geo=c('Ireland','Denmark'))

# Return the http response object for detailed Curl call response details
res <- bold_seq(taxon='Coelioxys', response=TRUE)
res$url
res$status_code
res$response_headers

## curl debugging
### You can do many things, including get verbose output on the curl
### call, and set a timeout
bold_seq(taxon='Coelioxys', verbose = TRUE)[1:2]
# bold_seqspec(taxon='Coelioxys', timeout_ms = 10)

## End(Not run)
```

---

bold_seqspec *Get BOLD specimen + sequence data.*

Description

Get BOLD specimen + sequence data.

Usage

```
bold_seqspec(
  taxon = NULL,
  ids = NULL,
  bin = NULL,
  container = NULL,
  institutions = NULL,
  researchers = NULL,
  geo = NULL,
  marker = NULL,
  response = FALSE,
  format = "tsv",
  sepfasta = FALSE,
  ...
)
```

Arguments

taxon	(character) Returns all records containing matching taxa. Taxa includes the ranks of phylum, class, order, family, subfamily, genus, and species.
ids	(character) Returns all records containing matching IDs. IDs include Sample IDs, Process IDs, Museum IDs and Field IDs.
bin	(character) Returns all records contained in matching BINs. A BIN is defined by a Barcode Index Number URI.
container	(character) Returns all records contained in matching projects or datasets. Containers include project codes and dataset codes
institutions	(character) Returns all records stored in matching institutions. Institutions are the Specimen Storing Site.
researchers	(character) Returns all records containing matching researcher names. Researchers include collectors and specimen identifiers.
geo	(character) Returns all records collected in matching geographic sites. Geographic sites includes countries and province/states.
marker	(character) Returns all records containing matching marker codes. See Details.
response	(logical) Note that response is the object that returns from the Curl call, useful for debugging, and getting detailed info on the API call.
format	(character) One of xml or tsv (default). tsv format gives back a data.frame object. xml gives back parsed xml as a
sepfasta	(logical) If TRUE, the fasta data is separated into a list with names matching the processid's from the data frame. Default: FALSE
...	Further args passed on to crul::verb-GET, main purpose being curl debugging

Value

Either a data.frame, parsed xml, a http response object, or a list with length two (a data.frame w/o nucleotide data, and a list with nucleotide data)

Large requests

Some requests can lead to errors. These often have to do with requesting data for a rank that is quite high in the tree, such as an Order, for example, Coleoptera. If your request is taking a long time, it's likely that something will go wrong on the BOLD server side, or we'll not be able to parse the result here in R because R can only process strings of a certain length. `bold` users have reported errors in which the resulting response from BOLD is so large that we could not parse it.

A good strategy for when you want data for a high rank is to do many separate requests for lower ranks within your target rank. You can do this manually, or use the function `taxize::downstream` to get all the names of a lower rank within a target rank. There's an example in the README (https://docs.ropensci.org/bold/#large-data)

If a request times out

This is likely because you're request was for a large number of sequences and the BOLD service timed out. You still should get some output, those sequences that were retrieved before the time out happened. As above, see the README (https://docs.ropensci.org/bold/#large-data) for an example of dealing with large data problems with this function.

Marker

Notes from BOLD on the `marker` param: "All markers for a specimen matching the search string will be returned. ie. A record with COI-5P and ITS will return sequence data for both markers even if only COI-5P was specified."

You will likely end up with data with markers that you did not request - just be sure to filter those out as needed.

References

http://v4.boldsystems.org/index.php/resources/api?type=webservices

Examples

```
## Not run:
bold_seqspec(taxon='Osmia')
bold_seqspec(taxon='Osmia', format='xml')
bold_seqspec(taxon='Osmia', response=TRUE)
res <- bold_seqspec(taxon='Osmia', sepfasta=TRUE)
res$fasta[1:2]
res$fasta['GBAH0293-06']

# records that match a marker name
res <- bold_seqspec(taxon="Melanogrammus aeglefinus", marker="COI-5P")

# records that match a geographic locality
res <- bold_seqspec(taxon="Melanogrammus aeglefinus", geo="Canada")

## curl debugging
### You can do many things, including get verbose output on the curl call,
### and set a timeout
head(bold_seqspec(taxon='Osmia', verbose = TRUE))
```

```
## timeout
# head(bold_seqspec(taxon='Osmia', timeout_ms = 1))

## End(Not run)
```

`bold_specimens` *Search BOLD for specimens.*

Description

Search BOLD for specimens.

Usage

```
bold_specimens(
  taxon = NULL,
  ids = NULL,
  bin = NULL,
  container = NULL,
  institutions = NULL,
  researchers = NULL,
  geo = NULL,
  response = FALSE,
  format = "tsv",
  ...
)
```

Arguments

`taxon`	(character) Returns all records containing matching taxa. Taxa includes the ranks of phylum, class, order, family, subfamily, genus, and species.
`ids`	(character) Returns all records containing matching IDs. IDs include Sample IDs, Process IDs, Museum IDs and Field IDs.
`bin`	(character) Returns all records contained in matching BINs. A BIN is defined by a Barcode Index Number URI.
`container`	(character) Returns all records contained in matching projects or datasets. Containers include project codes and dataset codes
`institutions`	(character) Returns all records stored in matching institutions. Institutions are the Specimen Storing Site.
`researchers`	(character) Returns all records containing matching researcher names. Researchers include collectors and specimen identifiers.
`geo`	(character) Returns all records collected in matching geographic sites. Geographic sites includes countries and province/states.
`response`	(logical) Note that response is the object that returns from the Curl call, useful for debugging, and getting detailed info on the API call.

format	(character) One of xml, json, tsv (default). tsv format gives back a data.frame object. xml gives back parsed XML as `xml_document` object. 'json' (JavaScript Object Notation) and 'dwc' (Darwin Core Archive) are supported in theory, but the JSON can be malformed, so we don't support that here, and the DWC option actually returns TSV.
...	Further args passed on to `crul::verb-GET`, main purpose being curl debugging

References

http://v4.boldsystems.org/index.php/resources/api?type=webservices

Examples

```
## Not run:
bold_specimens(taxon='Osmia')
bold_specimens(taxon='Osmia', format='xml')
bold_specimens(taxon='Osmia', response=TRUE)
res <- bold_specimens(taxon='Osmia', format='xml', response=TRUE)
res$url
res$status_code
res$response_headers

# More than 1 can be given for all search parameters
bold_specimens(taxon=c('Coelioxys','Osmia'))

## curl debugging
### These examples below take a long time, so you can set a timeout so that
### it stops by X sec
head(bold_specimens(taxon='Osmia', verbose = TRUE))
# head(bold_specimens(geo='Costa Rica', timeout_ms = 6))

## End(Not run)
```

---

bold_stats *Get BOLD stats*

Description

Get BOLD stats

Usage

```
bold_stats(
  taxon = NULL,
  ids = NULL,
  bin = NULL,
  container = NULL,
  institutions = NULL,
  researchers = NULL,
```

```
  geo = NULL,
  dataType = "drill_down",
  response = FALSE,
  ...
)
```

Arguments

taxon	(character) Returns all records containing matching taxa. Taxa includes the ranks of phylum, class, order, family, subfamily, genus, and species.
ids	(character) Returns all records containing matching IDs. IDs include Sample IDs, Process IDs, Museum IDs and Field IDs.
bin	(character) Returns all records contained in matching BINs. A BIN is defined by a Barcode Index Number URI.
container	(character) Returns all records contained in matching projects or datasets. Containers include project codes and dataset codes
institutions	(character) Returns all records stored in matching institutions. Institutions are the Specimen Storing Site.
researchers	(character) Returns all records containing matching researcher names. Researchers include collectors and specimen identifiers.
geo	(character) Returns all records collected in matching geographic sites. Geographic sites includes countries and province/states.
dataType	(character) one of "overview" or "drill_down" (default). "drill_down": a detailed summary of information which provides record counts by (BINs, Country, Storing Institution, Species). "overview": the total counts of (BINs, Countries, Storing Institutions, Orders, Families, Genus, Species)
response	(logical) Note that response is the object that returns from the Curl call, useful for debugging, and getting detailed info on the API call.
...	Further args passed on to crul::verb-GET, main purpose being curl debugging

References

http://v4.boldsystems.org/index.php/resources/api?type=webservices

Examples

```
## Not run:
x <- bold_stats(taxon='Osmia')
x$total_records
x$records_with_species_name
x$bins
x$countries
x$depositories
x$order
x$family
x$genus
x$species
```

```
# just get all counts
lapply(Filter(is.list, x), "[[", "count")

res <- bold_stats(taxon='Osmia', response=TRUE)
res$url
res$status_code
res$response_headers

# More than 1 can be given for all search parameters
bold_stats(taxon=c('Coelioxys','Osmia'))

## curl debugging
### These examples below take a long time, so you can set a timeout so that
### it stops by X sec
bold_stats(taxon='Osmia', verbose = TRUE)
# bold_stats(geo='Costa Rica', timeout_ms = 6)

## End(Not run)
```

`bold_tax_id` *Search BOLD for taxonomy data by BOLD ID.*

Description

Search BOLD for taxonomy data by BOLD ID.

Usage

```
bold_tax_id(
  id,
  dataTypes = "basic",
  includeTree = FALSE,
  response = FALSE,
  ...
)
```

Arguments

`id`	(integer) One or more BOLD taxonomic identifiers. required.
`dataTypes`	(character) Specifies the datatypes that will be returned. 'all' returns all data. 'basic' returns basic taxon information. 'images' returns specimen images.
`includeTree`	(logical) If `TRUE` (default: `FALSE`), returns a list containing information for parent taxa as well as the specified taxon.
`response`	(logical) Note that response is the object that returns from the Curl call, useful for debugging, and getting detailed info on the API call.
`...`	Further args passed on to `crul::verb-GET`, main purpose being curl debugging

References

http://v4.boldsystems.org/index.php/resources/api?type=taxonomy

See Also

bold_tax_name()

Examples

```
## Not run:
bold_tax_id(id=88899)
bold_tax_id(id=88899, includeTree=TRUE)
bold_tax_id(id=88899, includeTree=TRUE, dataTypes = "stats")
bold_tax_id(id=c(88899,125295))

## dataTypes parameter
bold_tax_id(id=88899, dataTypes = "basic")
bold_tax_id(id=88899, dataTypes = "stats")
bold_tax_id(id=88899, dataTypes = "images")
bold_tax_id(id=88899, dataTypes = "geo")
bold_tax_id(id=88899, dataTypes = "sequencinglabs")
bold_tax_id(id=88899, dataTypes = "depository")
bold_tax_id(id=c(88899,125295), dataTypes = "geo")
bold_tax_id(id=c(88899,125295), dataTypes = "images")

## Passing in NA
bold_tax_id(id = NA)
bold_tax_id(id = c(88899,125295,NA))

## get http response object only
bold_tax_id(id=88899, response=TRUE)
bold_tax_id(id=c(88899,125295), response=TRUE)

## curl debugging
bold_tax_id(id=88899, verbose = TRUE)

## End(Not run)
```

---

bold_tax_name *Search BOLD for taxonomy data by taxonomic name*

Description

Search BOLD for taxonomy data by taxonomic name

Usage

```
bold_tax_name(name, fuzzy = FALSE, response = FALSE, ...)
```

Arguments

name	(character) One or more scientific names. required.
fuzzy	(logical) Whether to use fuzzy search or not (default: FALSE)
response	(logical) Note that response is the object that returns from the Curl call, useful for debugging, and getting detailed info on the API call.
...	Further args passed on to crul::verb-GET, main purpose being curl debugging

Details

The dataTypes parameter is not supported in this function. If you want to use that parameter, get an ID from this function and pass it into bold_tax_id, and then use the dataTypes parameter.

References

http://v4.boldsystems.org/index.php/resources/api?type=taxonomy

See Also

bold_tax_id()

Examples

```
## Not run:
bold_tax_name(name='Diplura')
bold_tax_name(name='Osmia')
bold_tax_name(name=c('Diplura','Osmia'))
bold_tax_name(name=c("Apis","Puma concolor","Pinus concolor"))
bold_tax_name(name='Diplur', fuzzy=TRUE)
bold_tax_name(name='Osm', fuzzy=TRUE)

## get http response object only
bold_tax_name(name='Diplura', response=TRUE)
bold_tax_name(name=c('Diplura','Osmia'), response=TRUE)

## Names with no data in BOLD database
bold_tax_name("Nasiaeshna pentacantha")
bold_tax_name(name = "Cordulegaster erronea")
bold_tax_name(name = "Cordulegaster erronea", response=TRUE)

## curl debugging
bold_tax_name(name='Diplura', verbose = TRUE)

## End(Not run)
```

bold_trace *Get BOLD trace files*

Description

Get BOLD trace files

Usage

```
bold_trace(
  taxon = NULL,
  ids = NULL,
  bin = NULL,
  container = NULL,
  institutions = NULL,
  researchers = NULL,
  geo = NULL,
  marker = NULL,
  dest = NULL,
  overwrite = TRUE,
  progress = TRUE,
  ...
)

read_trace(x)
```

Arguments

taxon	(character) Returns all records containing matching taxa. Taxa includes the ranks of phylum, class, order, family, subfamily, genus, and species.
ids	(character) Returns all records containing matching IDs. IDs include Sample IDs, Process IDs, Museum IDs and Field IDs.
bin	(character) Returns all records contained in matching BINs. A BIN is defined by a Barcode Index Number URI.
container	(character) Returns all records contained in matching projects or datasets. Containers include project codes and dataset codes
institutions	(character) Returns all records stored in matching institutions. Institutions are the Specimen Storing Site.
researchers	(character) Returns all records containing matching researcher names. Researchers include collectors and specimen identifiers.
geo	(character) Returns all records collected in matching geographic sites. Geographic sites includes countries and province/states.
marker	(character) Returns all records containing matching marker codes.
dest	(character) A directory to write the files to

overwrite	(logical) Overwrite existing directory and file?
progress	(logical) Print progress or not. NOT AVAILABLE FOR NOW. HOPEFULLY WILL RETURN SOON.
...	Further args passed on to crul::verb-GET
x	Object to print or read.

References

http://v4.boldsystems.org/index.php/resources/api?type=webservices

Examples

```
## Not run:
# Use a specific destination directory
bold_trace(taxon='Bombus', geo='Alaska', dest="~/mytarfiles")

# Another example
# bold_trace(ids='ACRJP618-11', dest="~/mytarfiles")
# bold_trace(ids=c('ACRJP618-11','ACRJP619-11'), dest="~/mytarfiles")

# read file in
x <- bold_trace(ids=c('ACRJP618-11','ACRJP619-11'), dest="~/mytarfiles")
(res <- read_trace(x$ab1[2]))

# The progress dialog is pretty verbose, so quiet=TRUE is a nice touch,
# but not by default
# Beware, this one take a while
# x <- bold_trace(taxon='Osmia', quiet=TRUE)

if (requireNamespace("sangerseqR", quietly = TRUE)) {
 library("sangerseqR")
 primarySeq(res)
 secondarySeq(res)
 head(traceMatrix(res))
}

## End(Not run)
```

---

sequences *List of 3 nucleotide sequences to use in examples for the* bold_identify() *function*

Description

List of 3 nucleotide sequences to use in examples for the bold_identify() function

Details

Each sequence is a character string, of lengths 410, 600, and 696.

Index