# Package: bold (via r-universe)

October 24, 2024

**Version** 1.3.0

**Title** Interface to Bold Systems API

**Depends** R (>= 2.10)

**Suggests** sangerseqR, testthat, vcr (>= 0.5.4), tibble

**Imports** xml2, crul (>= 0.3.8), stringi, jsonlite, data.table, methods

**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.

**License** MIT + file LICENSE

**URL** https://docs.ropensci.org/bold/ (documentation), https://github.com/ropensci/bold (source)

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

**LazyData** yes

**Encoding** UTF-8

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.2.3

**X-schema.org-applicationCategory** Data Access

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

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

**Repository** https://ropensci.r-universe.dev

**RemoteUrl** https://github.com/ropensci/bold

**RemoteRef** master

**RemoteSha** 404fd118d87ddc75bcc6c075d03c734bd449f0bf

# Contents



---

`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 (Deprecated)
- `bold_tax_id2` - Get taxonomic names via BOLD identifiers (improved)
- `bold_identify` - Search for match given a COI sequence
- `bold_identify_parents` - Adds guessed parent ranks (Deprecated)

- bold_identify_taxonomy - Adds real parent ranks.

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", returnTibble = TRUE)
```

## 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 (-) |
| returnTibble | Whether the output should be a tibble or a data.frame. Default is TRUE, but verifies that the tibble package is installed, if it's not, it will be returned as data.frame. Since this package is only used in this function, doing this so it can be moved to suggested instead of dependency without breaking old scripts. |

## Value

a 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_get_attr *Get the error messages and parameters used for a request from a bold output.*

## Description

Get the error messages and parameters used for a request from a bold output.

Get the error messages from a bold output.

Get the parameters used for a request from a bold output.

## Usage

```
bold_get_attr(x)

bold_get_errors(x)

bold_get_params(x)
```

## Arguments

| | |
|---|---|
| x | Any object with an attribute "params". Usually the output of bold_tax_name or bold_tax_id2. |

## Value

A list of the attributes 'errors' and 'params' of the object.

The 'errors' attribute of the object.

The 'params' attribute of the object.

## Examples

```
## Not run:
x <- bold_tax_name(name=c("Apis","Felis","Pinus"), tax_division = "Animalia")
bold_get_errors(x)
y <- bold_tax_id2(id = c(88899999, 125295, NA_integer_), dataTypes = c("basic", "stats"))
bold_get_errors(y)

## End(Not run)
## Not run:
x <- bold_tax_name(name=c("Apis","Felis","Pinus"), tax_division = "Animalia")
bold_get_errors(x)
y <- bold_tax_id2(id = c(88899999, 125295, NA_integer_), dataTypes = c("basic", "stats"))
bold_get_errors(y)

## End(Not run)
## Not run:
x <- bold_tax_name(name=c("Apis","Felis","Pinus"), tax_division = "Animalia")
```

```
bold_get_params(x)
y <- bold_tax_id2(id = c(88899999, 125295, NA_integer_), dataTypes = c("basic", "stats"))
bold_get_params(y)

## 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 = c("COX1", "COX1_SPECIES", "COX1_SPECIES_PUBLIC", "COX1_L640bp"),
  response = FALSE,
  keepSeq = TRUE,
  ...
)
```

## Arguments

| | |
|---|---|
| `sequences` | (character) A vector or list of sequences to identify. Required. See Details. |
| `db` | (character) The database to match against, one of COX1 (default), COX1_SPECIES, COX1_SPECIES_PUBLIC, OR COX1_L640bp. 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. |
| `keepSeq` | (logical) If TRUE (default), returns each data.frame with an attribute 'sequence' containing sequence used to get those results. |
| `...` | Further args passed on to `verb-GET`, main purpose being curl debugging |

## Details

BOLD only allows one sequences per query. We internally `lapply` over the input values given to the sequences‘ parameter to search with one sequences 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 or list of (one per sequences) with the top specimen matches (up to 100) and their details. If the query fails, returns `NULL`. Each data.frame has the attributes `sequence` with the provided sequence to match (unless keepSeq is set to FALSE) and `errors` with the error message given from a failed request.

### db parameter options

- COX1 Every COI barcode record on BOLD with a minimum sequences 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 sequences length of 500bp. This includes many species represented by only one or two specimens as well as all species with interim taxonomy. Note : Sometimes it does return matches that don't have a species level identification. Will be checking with BOLD.
- COX1_SPECIES_PUBLIC All published COI records from BOLD and GenBank with a minimum sequences length of 500bp. This library is a collection of records from the published projects section of BOLD.
- OR COX1_L640bp Subset of the Species library with a minimum sequences length of 640bp and containing both public and private records. This library is intended for short sequences identification as it provides maximum overlap with short reads from the barcode region of COI.

### Named outputs

For a named output list, make sure to pass in a named list or vector to the `sequences` parameter. You can use `names<-` or `setNames` to set names on a list or vector of sequences.

### 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_taxonomy` *Add taxonomic parent names to a data set containing the process IDs of identified sequences.*

## Description

Add taxonomic parent names to a data set containing the process IDs of identified sequences.

## Usage

```
bold_identify_taxonomy(x, taxOnly = TRUE)

## S4 method for signature 'list'
bold_identify_taxonomy(x, taxOnly = TRUE)

## S4 method for signature 'matrix'
bold_identify_taxonomy(x, taxOnly = TRUE)

## S4 method for signature 'data.frame'
bold_identify_taxonomy(x, taxOnly = TRUE)

## S4 method for signature 'missing'
bold_identify_taxonomy(x, taxOnly = TRUE)
```

## Arguments

| | |
|---|---|
| `x` | A single data.frame or matrix, or a list of. Usually the output from a call to `bold_identify`. Required. |
| `taxOnly` | (logical) If TRUE (Default), only the taxonomic names and ids are added (equivalent format to the results of `bold_identify_parents` when `wide`is set to TRUE. If FALSE, also joins the rest of the data returned by `bold_specimens`. |

## Details

This function gets the process ids from the input data.frame(s) (ID column), then queries `bold_specimens` to get the sample information and adds it to the input data.frame(s).

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

## Value

a data.frame or a list of data.frames with added taxonomic classification.

## Examples

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

seqs_tax <- bold_identify_taxonomy(seqs)
head(seqs_tax[[1]])

x <- bold_seq(taxon = "Satyrium")
seqs <- bold_identify(x$sequence[1:2])
seqs_tax <- bold_identify_taxonomy(seqs)
seqs_tax

## 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) One or more taxonomic name. Optional. |
| ids | (character\|integer\|numeric) One or more IDs. Optional. IDs include Sample IDs, Process IDs, Museum IDs and Field IDs. |
| bin | (character) One or more Barcode Index Number URI. Optional. |
| container | (character) One or more project codes or dataset codes. Optional. |
| institutions | (character) One or more institution's name. Optional. Institutions are the Specimen Storing Site. |

| | |
|---|---|
| researchers | (character) One or more researcher names. Optional. Include collectors and specimen identifiers. |
| geo | (character) One or more geographic sites. Includes countries and province/states. |
| marker | (character) Returns all records containing matching marker codes. |
| response | (logical) Default : FALSE. If TRUE, returns the object from the Curl call. Useful for debugging and getting more detailed info on the API call. |
| ... | Further args passed on to crul::verb-GET, main purpose being curl debugging |

## Value

A data frame with each element as row and 5 columns for processid, identification, marker, accession, 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.

## Note

If using the taxon parameter with another parameter, if the taxon isn't found in the public database, it will act as if no taxon was specified and try to return all the data for the other specified parameter. You can make sure that the taxon you're looking up has public records with bold_stats.

## References

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

## Examples

```
## Not run:
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]

## 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,
```

```
  cleanData = FALSE,
  ...
)
```

## Arguments

| | |
|---|---|
| taxon | (character) One or more taxonomic name. Optional. |
| ids | (character\|integer\|numeric) One or more IDs. Optional. IDs include Sample IDs, Process IDs, Museum IDs and Field IDs. |
| bin | (character) One or more Barcode Index Number URI. Optional. |
| container | (character) One or more project codes or dataset codes. Optional. |
| institutions | (character) One or more institution's name. Optional. Institutions are the Specimen Storing Site. |
| researchers | (character) One or more researcher names. Optional. Include collectors and specimen identifiers. |
| geo | (character) One or more geographic sites. Includes countries and province/states. |
| marker | (character) Returns all records containing matching marker codes. See Details. |
| response | (logical) Default : FALSE. If TRUE, returns the object from the Curl call. Useful for debugging and getting more 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 list. |
| sepfasta | (logical) If TRUE, the fasta data is separated into a list with names matching the processid's for each records. Works with both 'tsv' and 'xml' format. Note: This means multiple sequences can have the same name if a process id has multiple sequences. Default: FALSE |
| cleanData | (logical) If TRUE, the cell values containing only duplicated values (ex : "COI-5P\|COI-5P\|COI-5P") will be reduce to one value ("COI-5P") and empty string will be change to NA. 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 of length two (data: a data.frame w/o nucleotide column, and fasta: a list of nucleotides with the processid as name)

## 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.

### Note

If using the `taxon` parameter with another parameter, if the `taxon` isn't found in the public database, it will act as if no `taxon` was specified and try to return all the data for the other specified parameter. You can make sure that the `taxon` you're looking up has public records with `bold_stats`.

### 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))
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",
  cleanData = FALSE,
  ...
)
```

## Arguments

| | |
|---|---|
| `taxon` | (character) One or more taxonomic name. Optional. |
| `ids` | (character\|integer\|numeric) One or more IDs. Optional. IDs include Sample IDs, Process IDs, Museum IDs and Field IDs. |
| `bin` | (character) One or more Barcode Index Number URI. Optional. |
| `container` | (character) One or more project codes or dataset codes. Optional. |
| `institutions` | (character) One or more institution's name. Optional. Institutions are the Specimen Storing Site. |
| `researchers` | (character) One or more researcher names. Optional. Include collectors and specimen identifiers. |
| `geo` | (character) One or more geographic sites. Includes countries and province/states. |
| `response` | (logical) Default : FALSE. If TRUE, returns the object from the Curl call. Useful for debugging and getting more 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. |
| `cleanData` | (logical) If `TRUE`, the cell values containing only duplicated values (ex : "COI-5P\|COI-5P\|COI-5P") will be reduce to one value ("COI-5P") and empty string will be change to NA. Default: `FALSE` |
| `...` | Further args passed on to `crul::verb-GET`, main purpose being curl debugging |

## Note

If using the `taxon` parameter with another parameter, if the `taxon` isn't found in the public database, it will act as if no `taxon` was specified and try to return all the data for the other specified parameter. You can make sure that the `taxon` you're looking up has public records with `bold_stats`.

## 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,
  simplify = FALSE,
  ...
)
```

## Arguments

| | |
|---|---|
| taxon | (character) One or more taxonomic name. Optional. |
| ids | (character\|integer\|numeric) One or more IDs. Optional. IDs include Sample IDs, Process IDs, Museum IDs and Field IDs. |
| bin | (character) One or more Barcode Index Number URI. Optional. |
| container | (character) One or more project codes or dataset codes. Optional. |
| institutions | (character) One or more institution's name. Optional. Institutions are the Specimen Storing Site. |
| researchers | (character) One or more researcher names. Optional. Include collectors and specimen identifiers. |
| geo | (character) One or more geographic sites. Includes countries and province/states. |
| dataType | (character) one of "drill_down"(default) or "overview". "drill_down": a detailed summary of information which provides record counts by BINs, Countries, Storing Institutions, Orders, Families, Genus, Species. "overview": the total record counts of BINs, Countries, Storing Institutions, Orders, Families, Genus, Species. The record counts include all gene markers, not only COI. To see the drill down of markers use bold_tax_id2 with "stats" as dataTypes. |
| response | (logical) Default : FALSE. If TRUE, returns the object from the Curl call. Useful for debugging and getting more detailed info on the API call. |
| simplify | (logical) whether the returned list should be simplified to a data.frame. See Details. |
| ... | Further args passed on to crul::verb-GET, main purpose being curl debugging |

## Value

By default, returns a nested list with the number of total records, the number of records with a species name, then for each of bins, countries, depositories, order, family, genus and species, the total count and the drill down of the records by up to 10 entities of that category. If simplify is set to TRUE, returns a list of length 2 : the overview data (number of total records, the number of records with a species name, and the total counts) simplified as a data.frame of 1 row and 9 columns and the drill_down data simplified to a one level list of data.frame. When dataType is set to "overview", returns a nested list with the number of total records, the number of records with a species name, and the total count for each of bins, countries, depositories, order, family, genus and species. If simplify is set to TRUE, returns a data.frame of 1 row and 9 columns.

## 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")

bold_stats(taxon='Osmia', dataType = "overview", simplified = TRUE)

x <- bold_stats(taxon='Osmia', simplified = TRUE)
x$overview
x$drill_down

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

## 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_id2` *Search BOLD for taxonomy data by BOLD ID.*

---

## Description

Search BOLD for taxonomy data by BOLD ID.

## Usage

```
bold_tax_id2(
  id,
```

```
  dataTypes = "basic",
  includeTree = FALSE,
  response = FALSE,
  ...
)
```

## Arguments

| | |
|---|---|
| id | (integer\|numeric\|character) One or more BOLD taxonomic identifiers. required. |
| dataTypes | (character) One or more BOLD data types: 'basic', 'stats', 'geo', 'images'/'img', 'sequencinglabs'/'labs', 'depository'/'depo', 'thirdparty'/'wiki' or 'all' (default: 'basic'). Specifies the information that will be returned, see details. |
| includeTree | (logical) If TRUE (default: FALSE), returns the information for parent taxa as well as the specified taxon, see details. |
| response | (logical) Default : FALSE. If TRUE, returns the object from the Curl call. Useful for debugging and getting more detailed info on the API call. |
| ... | Further args passed on to crul::verb-GET, main purpose being curl debugging |

## dataTypes

"basic" returns basic taxonomy info: includes taxid, taxon name, tax rank, tax division, parent taxid, parent taxon name. "stats" returns specimen and sequence statistics: includes public species count, public BIN count, public marker counts, public record count, specimen count, sequenced specimen count, barcode specimen count, species count, barcode species count. "geo" returns collection site information: includes country, collection site map. "images" returns specimen images: includes copyright information, image URL, image metadata. "sequencinglabs" returns sequencing labs: includes lab name, record count. "depository" returns specimen depositories: includes depository name, record count. "thirdparty" returns information from third parties: includes wikipedia_summary summary, wikipedia_summary URL. "all" returns all information: identical to specifying all data types at once.

## includeTree

When includeTree is set to true, for the dataTypes other than 'basic' the information of the parent taxa are identified by their taxonomic id only. To get their ranks and names too, make sure 'basic' is in dataTypes.

## References

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

## See Also

bold_tax_name, bold_get_attr, bold_get_errors, bold_get_params

## Examples

```
## Not run:
bold_tax_id2(id = 88899)
bold_tax_id2(id = 88899, includeTree = TRUE)
bold_tax_id2(id = 88899, includeTree = TRUE, dataTypes = "stats")
bold_tax_id2(id = c(88899, 125295))

## dataTypes parameter
bold_tax_id2(id = 88899, dataTypes = "basic")
bold_tax_id2(id = 88899, dataTypes = "stats")
bold_tax_id2(id = 88899, dataTypes = "images")
bold_tax_id2(id = 88899, dataTypes = "geo")
bold_tax_id2(id = 88899, dataTypes = "sequencinglabs")
bold_tax_id2(id = 88899, dataTypes = "depository")
bold_tax_id2(id = 88899, dataTypes = "thirdparty")
bold_tax_id2(id = 88899, dataTypes = "all")
bold_tax_id2(id = c(88899, 125295), dataTypes = c("basic", "geo"))
bold_tax_id2(id = c(88899, 125295), dataTypes = c("basic", "stats", "images"))

## Passing in NA
bold_tax_id2(id = NA)
bold_tax_id2(id = c(88899, 125295, NA))

## get http response object only
bold_tax_id2(id = 88899, response = TRUE)
bold_tax_id2(id = c(88899, 125295), response = TRUE)

## curl debugging
bold_tax_id2(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,
  tax_division = NULL,
  tax_rank = NULL,
  ...
)
```

## Arguments

| | |
|---|---|
| name | (character) One or more scientific names. required. |
| fuzzy | (logical) Whether to use fuzzy search or not (default: FALSE) |
| response | (logical) Default : FALSE. If TRUE, returns the object from the Curl call. Useful for debugging and getting more detailed info on the API call. |
| tax_division | (character) Taxonomic division to filter the results. |
| tax_rank | (character) Taxonomic rank to filter the results. |
| ... | 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.

## Note

The column 'specimenrecords' in the returned object represents the number of species records in BOLD's *Taxonomy Browser*, not the number of records in the *Public Data Portal*. To know the amount of public records available, use bold_stats.

## References

Taxonomy API: http://v4.boldsystems.org/index.php/resources/api?type=taxonomy Taxonomy Browser: https://www.boldsystems.org/index.php/TaxBrowser_Home Public Data Portal: https://www.boldsystems.org/index.php/Pub

## See Also

bold_tax_id, bold_get_attr, bold_get_errors, bold_get_params

## 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,
  ...
)

## S3 method for class 'boldtrace'
print(x, ...)

read_trace(x)

bold_read_trace(x)
```

## Arguments

| | |
|---|---|
| taxon | (character) One or more taxonomic name. Optional. |
| ids | (character\|integer\|numeric) One or more IDs. Optional. IDs include Sample IDs, Process IDs, Museum IDs and Field IDs. |
| bin | (character) One or more Barcode Index Number URI. Optional. |
| container | (character) One or more project codes or dataset codes. Optional. |
| institutions | (character) One or more institution's name. Optional. Institutions are the Specimen Storing Site. |
| researchers | (character) One or more researcher names. Optional. Include collectors and specimen identifiers. |

| | |
|---|---|
| geo | (character) One or more 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 verb-GET |
| x | (list or character) Either the boldtrace object returned from bold_trace or the path(s) of bold trace file(s). |

## References

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

## Examples

```
## Not run:
# Use a specific destination directory
bold_trace(taxon = "Bombus ignitus", geo = "Japan", 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]))
# read all files in
(res <- read_trace(x))

# 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)
```

## b_sepFasta *Separate sequences (fasta) from* bold_seqspec *output.*

### Description

Separate sequences (fasta) from bold_seqspec output.

### Usage

```
b_sepFasta(x, format = "tsv")
```

### Arguments

| | |
|---|---|
| x | (object) The output of a bold_seqspec call. |
| format | (character) The format used in the bold_seqspec call. One of 'tsv' (default) or 'xml'. |

### Value

A list of length two : the specimen data and the sequences list.

### Examples

```
## Not run:
res <- bold_seqspec(taxon='Osmia')
res <- b_sepFasta(res)
# (same as bold_seqspec(taxon='Osmia', sepFasta = TRUE))

## 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

### Format

list of length 3

### Details

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

# Index