Package ‘worrms’

June 28, 2019

Type Package

Title World Register of Marine Species (WoRMS) Client

Description Client for World Register of Marine Species
(<http://www.marinespecies.org/>). Includes functions for each of the API methods, including searching for names by name, date and common names, searching using external identifiers, fetching synonyms, as well as fetching taxonomic children and taxonomic classification.

Version 0.4.0

License MIT + file LICENSE

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

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

LazyData true

VignetteBuilder knitr

Encoding UTF-8

Language en-US

Imports crul (>= 0.6.0), tibble (>= 1.2), jsonlite (>= 1.1),
data.table

Suggests roxygen2 (>= 6.1.1), knitr, testthat, vcr (>= 0.2.0)

RoxygenNote 6.1.1

X-schema.org-applicationCategory Biology

X-schema.org-keywords biology, science, marine, API, web, api-client, worms, species

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 2019-06-28 06:00:04 UTC
R topics documented:

- `worrms-package` ........................................... 2
- `wm_attr_aphia` ............................................. 3
- `wm_attr_category` ......................................... 4
- `wm_attr_data` ................................................ 5
- `wm_attr_def` .................................................. 6
- `wm_children` .................................................. 7
- `wm_classification` .......................................... 8
- `wm_common_id` .............................................. 9
- `wm_distribution` ........................................... 10
- `wm_external` .............................................. 11
- `wm_id2name` ............................................. 12
- `wm_name2id` ............................................... 13
- `wm_ranks` .................................................. 14
- `wm_record` .................................................. 15
- `wm_records_common` ....................................... 16
- `wm_records_date` .......................................... 17
- `wm_records_name` .......................................... 17
- `wm_records_names` ........................................ 18
- `wm_records_rank` .......................................... 19
- `wm_records_taxamatch` .................................... 20
- `wm_record_by_external` ................................... 21
- `wm_sources` .............................................. 22
- `wm_synonyms` ............................................ 23

Index 24

---

**Description**

World Register of Marine Species Client

**Fail behavior**

The WoRMS REST API doesn’t have sophisticated error messaging, so most errors will result in a (204) - No Content or in (400) - Bad Request.

Because WoRMS doesn’t do comprehensive error reporting, we do a fair amount of checking user inputs to help prevent errors that will be meaningless to the user. Let us know if we can improve on this.

**Author(s)**

Scott Chamberlain <myrmecocystus@gmail.com>
### Description

Get AphiaIDs by attribute definition ID

### Usage

```r
wm_attr_aphia(id, offset = 1, ...)

wm_attr_aphia_(id = NULL, name = NULL, ...)
```

### Arguments

- **id** (numeric/integer) A attribute ID. For `wm_attr_aphia` it’s required and must be `length(id) == 1`, for `wm_attr_aphia_` it’s optional and can be `length(id) >= 1`
- **offset** (integer) record to start at. default: 1
- **...** named curl options. see `curl::curl_options`
- **name** (character) one or more taxonomic names. optional

### Value

A tibble/data.frame. when using underscore method, outputs from each input are binded together, but can be split by id column

### Singular vs. plural

Of the two sister functions, the one without the underscore is the original function that wraps the relevant WoRMS API method - and only accepts one thing (i.e., name or AphiaID) per request.

The sister function with the underscore at the end is the plural version, accepting more than one input. Internally this function loops over the non-underscore method, and labels output (whether it’s a list or data.frame rows) with the input names or IDs so that you can easily parse output by your inputs.

### Examples

```r
## Not run:
wm_attr_aphia(id = 7)
wm_attr_aphia(id = 4)
wm_attr_aphia(id = 4, offset = 50)
wm_attr_aphia_(id = c(7, 2))

## End(Not run)
```
**wm_attr_category**

Get attributes grouped by a CategoryID

---

**Description**

Get attributes grouped by a CategoryID

**Usage**

```r
wm_attr_category(id, ...)
wm_attr_category_(id = NULL, name = NULL, ...)
```

**Arguments**

- `id` (numeric/integer) a CategoryID. For `wm_attr_category` it's required and must be `length(id) == 1`, for `wm_attr_category_` it's optional and can be `length(id) >= 1`
- `...` named curl options. see `curl::curl_options`
- `name` (character) one or more taxonomic names. optional

**Value**

A tibble/data.frame. when using underscore method, outputs from each input are binded together, but can be split by `id` column

**Singular vs. plural**

Of the two sister functions, the one without the underscore is the original function that wraps the relevant WoRMS API method - and only accepts one thing (i.e., name or AphiaID) per request.

The sister function with the underscore at the end is the plural version, accepting more than one input. Internally this function loops over the non-underscore method, and labels output (whether it’s a list or data.frame rows) with the input names or IDs so that you can easily parse output by your inputs.

**Examples**

```r
## Not run:
wm_attr_category(id = 7)
wm_attr_category(id = 2)
wm_attr_category_(id = c(7, 2))
## End(Not run)
```
### Description

Get attribute data by AphiaID

### Usage

```r
wm_attr_data(id, include_inherited = FALSE, ...)
```

```r
wm_attr_data_(id = NULL, name = NULL, ...)
```

### Arguments

- **id** (numeric/integer) an AphiaID. For `wm_attr_data` it's required and must be `length(id) == 1`, for `wm_attr_data_` it's optional and can be `length(id) >= 1`
- **include_inherited** (logical) Include attributes inherited from its parent taxon. Default: FALSE
- **...** named curl options. see `curl::curl_options`
- **name** (character) one or more taxonomic names. optional

### Value

A tibble/data.frame. when using underscore method, outputs from each input are binded together, but can be split by id column

### Singular vs. plural

Of the two sister functions, the one without the underscore is the original function that wraps the relavant WoRMS API method - and only accepts one thing (i.e., name or AphiaID) per request.

The sister function with the underscore at the end is the plural version, accepting more than one input. Internally this function loops over the non-underscore method, and labels output (whether it's a list or data.frame rows) with the input names or IDs so that you can easily parse output by your inputs.

### Examples

```r
## Not run:
wm_attr_data(id = 127160)
wm_attr_data(id = 126436)
wm_attr_data_(id = c(127160, 126436))

## End(Not run)
```
wm_attr_def

Get attribute definition by ID

Description

Get attribute definition by ID

Usage

wm_attr_def(id, include_inherited = FALSE, ...)

wm_attr_def_(id = NULL, name = NULL, ...)

Arguments

id (numeric/integer) an attribute ID. For wm_attr_def it’s required and must be length(id) == 1, for wm_attr_def_ it’s optional and can be length(id) >= 1

include_inherited (logical) Include attributes inherited from its parent taxon. Default: FALSE

... named curl options. see curl::curl_options

name (character) one or more taxonomic names. optional

Value

A tibble/data.frame. when using underscore method, outputs from each input are binded together, but can be split by id column

Singular vs. plural

Of the two sister functions, the one without the underscore is the original function that wraps the relevant WoRMS API method - and only accepts one thing (i.e., name or AphiaID) per request.

The sister function with the underscore at the end is the plural version, accepting more than one input. Internally this function loops over the non-underscore method, and labels output (whether it’s a list or data.frame rows) with the input names or IDs so that you can easily parse output by your inputs.

Examples

## Not run:
wm_attr_def(id = 1)
wm_attr_def(id = 4)
wm_attr_def(id = 4, include_inherited = TRUE)

wm_attr_def_(id = c(4, 1))

## End(Not run)
wm_children

Get children for an AphiaID

Description
Get children for an AphiaID

Usage
wm_children(id, marine_only = TRUE, offset = 1, ...)

wm_children_(id = NULL, name = NULL, marine_only = TRUE,
offset = 1, ...)

Arguments
id (numeric/integer) an AphiaID. For wm_children it's required and must be length(id) == 1,
for wm_children_ it's optional and can be length(id) >= 1
marine_only (logical) marine only or not. default: TRUE
offset (integer) record to start at. default: 1
... named curl options. see curl::curl_options
name (character) one or more taxonomic names. optional

Value
A tibble/data.frame. when using underscore method, outputs from each input are binded together,
but can be split by id column

Singular vs. plural
Of the two sister functions, the one without the underscore is the original function that wraps the
relavant WoRMS API method - and only accepts one thing (i.e., name or AphiaID) per request.
The sister function with the underscore at the end is the plural version, accepting more than one
input. Internally this function loops over the non-underscore method, and labels output (whether
it's a list or data.frame rows) with the input names or IDs so that you can easily parse output by
your inputs.

Examples
## Not run:
wm_children(343613)
wm_children(id = 105706)
wm_children(id = 105706, FALSE)
wm_children(id = 105706, offset = 5)

# plural version, via id or name
wm_children_(id = c(105706, 343613))
wm_classification

wm_classification(name = c('Mesodesma', 'Leucophaeus'))

## End(Not run)

---

wm_classification Get classification for an AphiaID

**Description**

Get classification for an AphiaID

**Usage**

```r
wm_classification(id, ...)
wm_classification_(id = NULL, name = NULL, ...)
```

**Arguments**

- `id` (numeric/integer) an AphiaID. For `wm_children` it's required and must be `length(id) == 1`, for `wm_children_` it's optional and can be `length(id) >= 1`
- `...` named curl options. see `curl::curl_options`
- `name` (character) one or more taxonomic names. optional

**Value**

A tibble/data.frame. when using underscore method, outputs from each input are binded together, but can be split by `id` column

**Singular vs. plural**

Of the two sister functions, the one without the underscore is the original function that wraps the relevant WoRMS API method - and only accepts one thing (i.e., name or AphiaID) per request.

The sister function with the underscore at the end is the plural version, accepting more than one input. Internally this function loops over the non-underscore method, and labels output (whether it's a list or data.frame rows) with the input names or IDs so that you can easily parse output by your inputs.

**Examples**

```r
## Not run:
wm_classification(id = 105706)
wm_classification(id = 126436)

wm_classification(254967)
wm_classification(344089)

# plural version, via id or name
```
wm_common_id

wm_classification(id = c(254967, 344089))
wm_classification(name = c('Platanista gangetica', 'Leucophaeus scoresbii'))

## End(Not run)

wm_common_id

Get vernacular names from an AphialID

Description
Get vernacular names from an AphialID

Usage
wm_common_id(id, ...)
wm_common_id_(id = NULL, name = NULL, ...)

Arguments

id  (numeric/integer) an AphialID. For wm_common_id it's required and must be length(id) == 1, for wm_common_id_ it's optional and can be length(id) >= 1

... named curl options. see curl::curl_options

name  (character) one or more taxonomic names. optional

Value
A tibble/data.frame. when using underscore method, outputs from each input are binded together, but can be split by id column

Singular vs. plural

Of the two sister functions, the one without the underscore is the original function that wraps the relevant WoRMS API method - and only accepts one thing (i.e., name or AphialID) per request.

The sister function with the underscore at the end is the plural version, accepting more than one input. Internally this function loops over the non-underscore method, and labels output (whether it's a list or data.frame rows) with the input names or IDs so that you can easily parse output by your inputs.

Examples

## Not run:
wm_common_id(id = 105706)
wm_common_id(id = 156806)
wm_common_id(id = 397065)

wm_common_id_(id = c(105706, 156806, 397065))
nms <- c("Rhincodontidae", "Mesodesma deauratum", "Cryptomya californica")
wm_distribution

wm_common_id_(name = nms)

## End(Not run)

wm_distribution

Get distribution data by AphiaID

Description

Get distribution data by AphiaID

Usage

wm_distribution(id, ...)

wm_distribution_(id = NULL, name = NULL, ...)

Arguments

id (numeric/integer) an AphiaID. For `wm_distribution` it's required and must be
length(id) == 1, for `wm_distribution_` it's optional and can be length(id) >= 1

... named curl options. see `curl::curl_options`

name (character) one or more taxonomic names. optional

Value

A tibble/data.frame. when using underscore method, outputs from each input are binded together,
but can be split by id column

Singular vs. plural

Of the two sister functions, the one without the underscore is the original function that wraps the
relavant WoRMS API method - and only accepts one thing (i.e., name or AphiaID) per request.
The sister function with the underscore at the end is the plural version, accepting more than one
input. Internally this function loops over the non-underscore method, and labels output (whether
it’s a list or data.frame rows) with the input names or IDs so that you can easily parse output by
your inputs.

Examples

## Not run:
wm_distribution(id = 156806)
wm_distribution(id = 126436)

wm_distribution_(id = c(156806, 126436))

## End(Not run)
wm_external

Get an external ID via an AphiaID

Description

Get an external ID via an AphiaID

Usage

wm_external(id, type = "tsn", ...)

wm_external_(id = NULL, name = NULL, type = "tsn", ...)

Arguments

id (numeric/integer) an AphiaID. For \texttt{wm\_external} it’s required and must be length(id) \(\geq 1\), for \texttt{wm\_external\_} it’s optional and can be length(id) \(\geq 1\)

\texttt{type} (character) the type of external id. one of: tsn, bold, dyntaxa, eol, fishbase, iucn, lsid, ncbi, gisd. default: tsn

\ldots named curl options. see \texttt{curl::curl\_options}

\texttt{name} (character) one or more taxonomic names. optional

Value

An integer that is the ID. When using underscore method, a list, named by the input IDs

Singular vs. plural

Of the two sister functions, the one without the underscore is the original function that wraps the relavant WoRMS API method - and only accepts one thing (i.e., name or AphiaID) per request.

The sister function with the underscore at the end is the plural version, accepting more than one input. Internally this function loops over the non-underscore method, and labels output (whether it’s a list or data.frame rows) with the input names or IDs so that you can easily parse output by your inputs.

Examples

## Not run:
# by default, get a TSN (an ITIS code)
wm_external(id = 1080)

## get many
wm_external_(id = c(1080, 126436))

# BOLD code
wm_external(id = 278468, type = "bold")
# NCBI code
wm_external(id = 278468, type = "ncbi")

# fishbase code
wm_external(id = 278468, type = "fishbase")

# curl options
library(curl)
wm_external(id = 105706, verbose = TRUE)

## End(Not run)

---

### `wm_id2name`

Get taxonomic name for an AphiaID

#### Description

Get taxonomic name for an AphiaID

#### Usage

wm_id2name(id, ...)

wm_id2name_(id, ...)

#### Arguments

- **id** *(numeric/integer)* an AphiaID, required. For `wm_id2name` must be `length(id) == 1`, but for `wm_id2name_` can be `length(id) >= 1`
- **...** named curl options. see `curl::curl_options`

#### Value

An character string that is the taxonomic name. When using underscore method, a list, named by the input IDs

#### Singular vs. plural

Of the two sister functions, the one without the underscore is the original function that wraps the relevant WoRMS API method - and only accepts one thing (i.e., name or AphiaID) per request.

The sister function with the underscore at the end is the plural version, accepting more than one input. Internally this function loops over the non-underscore method, and labels output (whether it's a list or data.frame rows) with the input names or IDs so that you can easily parse output by your inputs.
wm_name2id

Examples

```r
## Not run:
wm_id2name(id = 105706)
wm_id2name_(id = c(105706, 126436))

## End(Not run)
```

---

**wm_name2id**

*Get AphiaID from a taxonomic name*

Description

Get AphiaID from a taxonomic name

Usage

```r
wm_name2id(name, ...)
wm_name2id_(name, ...)
```

Arguments

- `name` (character) a taxonomic name, required. For `wm_name2id` must be `length(name) == 1`, but for `wm_name2id_` can be `length(name) >= 1`
- `...` named curl options. see `curl::curl_options`

Value

An integer that is the AphiaID. When using underscore method, a list, named by the input names

Singular vs. plural

Of the two sister functions, the one without the underscore is the original function that wraps the relevant WoRMS API method - and only accepts one thing (i.e., name or AphiaID) per request.

The sister function with the underscore at the end is the plural version, accepting more than one input. Internally this function loops over the non-underscore method, and labels output (whether it’s a list or data.frame rows) with the input names or IDs so that you can easily parse output by your inputs.

Examples

```r
## Not run:
wm_name2id(name = "Rhincodon")
wm_name2id_(name = c("Rhincodon", "Gadus morhua"))

## End(Not run)
```
wm_ranks

*Get taxonomic ranks by their identifier*

**Description**

Get taxonomic ranks by their identifier

**Usage**

```r
wm_ranks_id(rank_id, id = NULL, offset = 1, ...)
wm_ranks_name(rank_name, id = NULL, offset = 1, ...)
```

**Arguments**

- `rank_id` (numeric/integer) a rank identifier. length==1
- `id` an AphiaID. length==1
- `offset` (integer) record to start at. default: 1
- `...` named curl options. see curl::curl_options
- `rank_name` (character) a rank name. length==1

**Value**

A tibble/data.frame

**Examples**

```r
## Not run:
wm_ranks_id(220)
wm_ranks_id(180)
wm_ranks_id(180, id = 4)

wm_ranks_name("genus")
wm_ranks_name("genus", id = 4)

## End(Not run)
```
wm_record

Get complete AphiaRecord for an AphiaID

Description
Get complete AphiaRecord for an AphiaID

Usage
wm_record(id, ...)

wm_record_(id = NULL, name = NULL, ...)

Arguments
id  (numeric/integer) an AphiaID. For wm_record it's required and must be length(id) == 1, for wm_record_ it's optional and can be length(id) >= 1

... named curl options. see curl::curl_options

name  (character) one or more taxonomic names. optional

Value
A named list. When using underscore method, each output is named by the input ID, and can be separated by the list names

Singular vs. plural
Of the two sister functions, the one without the underscore is the original function that wraps the relevant WoRMS API method - and only accepts one thing (i.e., name or AphiaID) per request.
The sister function with the underscore at the end is the plural version, accepting more than one input. Internally this function loops over the non-underscore method, and labels output (whether it’s a list or data.frame rows) with the input names or IDs so that you can easily parse output by your inputs.

Note
wm_record_ is defunct, wm_record can do plural requests now

Examples
## Not run:
wm_record(id = 105706)
wm_record(id = c(105706, 126436))
wm_record_(id = c(105706, 126436))

## End(Not run)
wm_records_common  
Get records by vernacular name, optional fuzzy matching

Description
Get records by vernacular name, optional fuzzy matching

Usage
wm_records_common(name, fuzzy = FALSE, offset = 1, ...)
wm_records_common_(name, fuzzy = FALSE, offset = 1, ...)

Arguments

name  (character) a species common name. required. For `wm_records_common` must be `length(name) == 1` ; for `wm_records_common_` can be `length(name) >= 1`

fuzzy  (logical) fuzzy search. default: FALSE

offset  (integer) record to start at. default: 1

...  named curl options. see `curl::curl_options`

Value
A tibble/data.frame. when using underscore method, outputs from each input are binded together, but can be split by `id` column

Singular vs. plural
Of the two sister functions, the one without the underscore is the original function that wraps the relevant WoRMS API method - and only accepts one thing (i.e., name or AphiaID) per request.
The sister function with the underscore at the end is the plural version, accepting more than one input. Internally this function loops over the non-underscore method, and labels output (whether it’s a list or data.frame rows) with the input names or IDs so that you can easily parse output by your inputs.

Examples

```r
## Not run:
wm_records_common(name = 'dolphin')
wm_records_common(name = 'clam')

wm_records_common_(name = c('dolphin', 'clam'))

wm_records_common(name = 'dolphin', fuzzy = TRUE)
wm_records_common(name = 'clam', fuzzy = TRUE, offset = 5)

## End(Not run)
```
**wm_records_date**

**Get records by date**

**Description**

Get records by date

**Usage**

```r
wm_records_date(start_date, end_date = NULL, marine_only = TRUE, offset = 1, ...)
```

**Arguments**

- `start_date` (character) start date. required.
- `end_date` (character) end date. optional
- `marine_only` (logical) marine only or not. default: TRUE
- `offset` (integer) record to start at. default: 1
- `...` named curl options. see `curl::curl_options`

**Value**

A tibble/data.frame

**Examples**

```r
## Not run:
a_date <- format(Sys.Date() - 1, "%Y-%m-%dT%H:%M:%S+00:00")
wm_records_date(a_date)

## End(Not run)
```

---

**wm_records_name**

**Get records by single name, optional fuzzy matching**

**Description**

Get records by single name, optional fuzzy matching

**Usage**

```r
wm_records_name(name, fuzzy = TRUE, marine_only = TRUE, offset = 1, ...
```

---
Arguments

- **name** (character) a taxonomic name, required.
- **fuzzy** (logical) fuzzy search. default: TRUE
- **marine_only** (logical) marine only or not. default: TRUE
- **offset** (integer) record to start at. default: 1

... named curl options. see curl::curl_options

Value

A tibble/data.frame

Note

there is no underscore method like other functions in this package as there is already a plural version: `wm_records_names()`

Examples

```r
## Not run:
wm_records_name(name = 'Leucophaeus')
wm_records_name(name = 'Leucophaeus', fuzzy = FALSE)
wm_records_name(name = 'Platanista', marine_only = FALSE)
wm_records_name(name = 'Platanista', marine_only = FALSE, offset = 5)

## End(Not run)
```

---

**wm_records_names**

*Get records for one or more taxonomic name(s)*

Description

Get records for one or more taxonomic name(s)

Usage

`wm_records_names(name, fuzzy = FALSE, marine_only = TRUE, ...)`

Arguments

- **name** (character) start date. required.
- **fuzzy** (logical) fuzzy search. default: FALSE
- **marine_only** (logical) marine only or not. default: TRUE

... named curl options. see curl::curl_options
wm_records_rank

Description

Get AphiaRecords for a given taxonRankID

Usage

wm_records_rank(rank_id, id = NULL, offset = 1, ...)

Arguments

- rank_id (numeric/integer) a rank id
- id (character) a single AphiaID
- offset (integer) record to start at. default: 1
  ...

Value

A tibble/data.frame. when using underscore method, outputs from each input are binded together, but can be split by id column

Singular vs. plural

Of the two sister functions, the one without the underscore is the original function that wraps the relevant WoRMS API method - and only accepts one thing (i.e., name or AphiaID) per request.

The sister function with the underscore at the end is the plural version, accepting more than one input. Internally this function loops over the non-underscore method, and labels output (whether it’s a list or data.frame rows) with the input names or IDs so that you can easily parse output by your inputs.
wm_records_taxamatch

**Examples**

```r
## Not run:
wm_records_rank(rank_id = 180, id = 106776)
wm_records_rank(rank_id = 180, id = 106776, offset = 50)

## End(Not run)
```

wm_records_taxamatch \*Get records for one or more taxonomic name(s) using the TAXAMATCH fuzzy matching algorithm*

**Description**

Get records for one or more taxonomic name(s) using the TAXAMATCH fuzzy matching algorithm

**Usage**

```r
wm_records_taxamatch(name, marine_only = TRUE, ...)
```

**Arguments**

- `name` (character) taxon name. required.
- `marine_only` (logical) marine only or not. default: TRUE
- `...` named curl options. see curl::curl_options

**Value**

A list of tibble’s/data.frame’s, one for each of the input names

**Note**

there is no underscore method like other functions in this package as this function already accepts many names

**Examples**

```r
## Not run:
wm_records_taxamatch(name = 'Leucophaeus')
wm_records_taxamatch(name = c('Leucophaeus', 'Coryphaena'))

## End(Not run)
```
wm_record_by_external  Get record by external ID

Description

Get record by external ID

Usage

```r
wm_record_by_external(id, type = "tsn", ...)
wm_record_by_external_(id = NULL, name = NULL, type = "tsn", ...)
```

Arguments

- **id** (numeric/integer) an AphiaID. For `wm_record_by_external` it's required and must be `length(id) == 1`, for `wm_record_by_external_` it's optional and can be `length(id) >= 1`
- **type** (character) the type of external id. one of: tsn, bold, dyntaxa, eol, fishbase, iucn, lsid, ncbi, gisd. default: tsn
- **...** named curl options. see `curl::curl_options`
- **name** (character) one or more taxonomic names. optional

Value

A named list. When using underscore method, each output is named by the input ID, and can be separated by the list names

Singular vs. plural

Of the two sister functions, the one without the underscore is the original function that wraps the relevant WoRMS API method - and only accepts one thing (i.e., name or AphiaID) per request.

The sister function with the underscore at the end is the plural version, accepting more than one input. Internally this function loops over the non-underscore method, and labels output (whether it's a list or data.frame rows) with the input names or IDs so that you can easily parse output by your inputs.

Examples

```r
## Not run:
wm_record_by_external(id = 85257)
wm_record_by_external(id = 159854)

wm_record_by_external_(id = c(85257, 159854))

## End(Not run)
```
**wm_sources**

*Get sources for an AphiaID*

### Description

Get sources for an AphiaID

### Usage

```r
wm_sources(id, ...)

wm_sources_(id = NULL, name = NULL, ...)
```

### Arguments

- **id**
  
  (numeric/integer) an AphiaID. For `wm_sources` it's required and must be `length(id) == 1`,
  for `wm_sources_` it's optional and can be `length(id) >= 1`

- **...**

  named curl options. see `curl::curl_options`

- **name**

  (character) one or more taxonomic names. optional

### Value

A tibble/data.frame. when using underscore method, outputs from each input are binded together,
but can be split by id column

### Singular vs. plural

Of the two sister functions, the one without the underscore is the original function that wraps
the relevant WoRMS API method - and only accepts one thing (i.e., name or AphiaID) per request.

The sister function with the underscore at the end is the plural version, accepting more than one
input. Internally this function loops over the non-underscore method, and labels output (whether
it's a list or data.frame rows) with the input names or IDs so that you can easily parse output by
your inputs.

### Examples

```r
## Not run:
wm_sources(id = 105706)
wm_sources_(id = 105706)
wm_sources_(id = c(105706, 126436))
wm_sources_(name = c("Rhincodontidae", "Gadus morhua"))
## End(Not run)
```
Get synonyms for an AphiaID

Description

Get synonyms for an AphiaID

Usage

wm_synonyms(id, offset = 1, ...)

wm_synonyms_(id = NULL, name = NULL, ...)

Arguments

id (numeric/integer) an AphiaID. For wm_synonyms it’s required and must be length(id) == 1, for wm_synonyms_ it’s optional and can be length(id) >= 1
offset (integer) record to start at. default: 1
... named curl options. see curl::curl_options
name (character) one or more taxonomic names. optional

Value

A tibble/data.frame. when using underscore method, outputs from each input are binded together, but can be split by id column

Singular vs. plural

Of the two sister functions, the one without the underscore is the original function that wraps the relavant WoRMS API method - and only accepts one thing (i.e., name or AphiaID) per request.

The sister function with the underscore at the end is the plural version, accepting more than one input. Internally this function loops over the non-underscore method, and labels output (whether it’s a list or data.frame rows) with the input names or IDs so that you can easily parse output by your inputs.

Examples

```r
## Not run:
wm_synonyms(id = 105706)
wm_synonyms_(id = 105706)
wm_synonyms(id = 126436)
wm_synonyms(id = 126436, offset = 10)
wm_synonyms_(id = c(105706, 126436))

## End(Not run)
```
Index

*Topic package
   worrms-package, 2

wm_attr_aphia, 3
wm_attr_aphia_(wm_attr_aphia), 3
wm_attr_category, 4
wm_attr_category_(wm_attr_category), 4
wm_attr_data, 5
wm_attr_data_(wm_attr_data), 5
wm_attr_def, 6
wm_attr_def_(wm_attr_def), 6
wm_children, 7
wm_children_(wm_children), 7
wm_classification, 8
wm_classification_(wm_classification), 8
wm_common_id, 9
wm_common_id_(wm_common_id), 9
wm_distribution, 10
wm_distribution_(wm_distribution), 10
wm_external, 11
wm_external_(wm_external), 11
wm_id2name, 12
wm_id2name_(wm_id2name), 12
wm_name2id, 13
wm_name2id_(wm_name2id), 13
wm_ranks, 14
wm_ranks_id (wm_ranks), 14
wm_ranks_name (wm_ranks), 14
wm_record, 15
wm_record (wm_record), 15
wm_record_by_external, 21
wm_record_by_external_ (wm_record_by_external), 21
wm_records_common, 16
wm_records_common_(wm_records_common), 16
wm_records_date, 17
wm_records_name, 17
wm_records_name(), 19
wm_records_names, 18
wm_records_names(), 18
wm_records_rank, 19
wm_records_taxamatch, 20
wm_sources, 22
wm_sources_(wm_sources), 22
wm_synonyms, 23
wm_synonyms_(wm_synonyms), 23
worrms (worrms-package), 2
worrms-package, 2