

# Package ‘rfishbase’

June 7, 2021

**Title** R Interface to 'FishBase'

**Description** A programmatic interface to 'FishBase', re-written based on an accompanying 'RESTful' API. Access tables describing over 30,000 species of fish, their biology, ecology, morphology, and more. This package also supports experimental access to 'SeaLifeBase' data, which contains nearly 200,000 species records for all types of aquatic life not covered by 'FishBase.'

**Version** 3.1.8

**Encoding** UTF-8

**License** CC0

**URL** <https://docs.ropensci.org/rfishbase/>,  
<https://github.com/ropensci/rfishbase>

**BugReports** <https://github.com/ropensci/rfishbase/issues>

**LazyData** true

**Depends** R (>= 4.0)

**Imports** dplyr, purrr, methods, utils, memoise, readr, rlang, magrittr,  
stringr, gh, DBI, arkdb (>= 0.0.12), tools, curl, dbplyr,  
progress, RSQLite

**Suggests** testthat, rmarkdown, knitr, covr, spelling

**VignetteBuilder** knitr

**RoxygenNote** 7.1.1

**Language** en-US

**NeedsCompilation** no

**Author** Carl Boettiger [cre, aut] (<<https://orcid.org/0000-0002-1642-628X>>),  
Scott Chamberlain [aut] (<<https://orcid.org/0000-0003-1444-9135>>),  
Duncan Temple Lang [aut],  
Peter Wainwright [aut],  
Kevin Cazelles [ctb] (<<https://orcid.org/0000-0001-6619-9874>>)

**Maintainer** Carl Boettiger <cboettig@gmail.com>

**Repository** CRAN

**Date/Publication** 2021-06-07 17:20:02 UTC

**R topics documented:**

rfishbase-package	3
available_releases	3
brains	4
common_names	5
common_to_sci	6
country	7
countrysub	8
countrysubref	9
c_code	9
default_db	10
diet	11
diet_items	13
distribution	14
docs	15
ecology	16
ecosystem	17
estimate	18
faoareas	19
fecundity	20
fishbase	21
fooditems	21
genetics	22
introductions	23
larvae	24
length_freq	25
length_length	26
length_weight	28
list_fields	29
load_taxa	30
maturity	31
morphology	32
morphometrics	33
occurrence	34
oxygen	34
popchar	35
popgrowth	36
popqb	37
predators	38
ration	39
references	40
reproduction	41
sealifebase	42
spawning	43
species	44
species_by_ecosystem	45
species_fields	46

species_list . . . . .	46
species_names . . . . .	47
speed . . . . .	48
stocks . . . . .	49
swimming . . . . .	50
synonyms . . . . .	51
validate_names . . . . .	52

## Index 53

rfishbase-package      *The new R interface to Fishbase, v2.0*

### Description

A programmatic interface to FishBase, re-written based on an accompanying 'RESTful' API. Access tables describing over 30,000 species of fish, their biology, ecology, morphology, and more. This package also supports experimental access to SeaLifeBase data, which contains nearly 200,000 species records for all types of aquatic life not covered by FishBase.'

### Author(s)

Carl Boettiger <carl@ropensci.org>  
 Scott Chamberlain <scott@ropensci.org>

available\_releases      *List available releases*

### Description

List available releases

### Usage

```
available_releases()
```

### Details

Lists all available releases (year.month format). To use a specific release, set the desired release using 'options(FISHBASE\_VERSION=)', as shown in the examples. Otherwise, rfishbase will use the latest available version if this option is unset. NOTE: it will be necessary to clear the cache with 'clear\_cache()' or by restarting the R session with a fresh environment.

### Examples

```
available_releases()
options(FISHBASE_VERSION="19.04")
## unset
options(FISHBASE_VERSION=NULL)
```

---

 brains
 

---

*brains***Description**

brains

**Usage**

```

brains(
  species_list = NULL,
  fields = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)

```

**Arguments**

species_list	A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.
fields	a character vector specifying which fields (columns) should be returned. By default, all available columns recognized by the parser are returned. Mostly for backwards compatibility as users can subset by column later
server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
version	a version string for the database, will default to the latest release. see [get_releases()] for details.
db	the
...	unused; for backwards compatibility only

**Value**

a table of species brains

**Examples**

```

## Not run:
brains("Oreochromis niloticus")

## End(Not run)

```

---

common_names	<i>common names</i>
--------------	---------------------

---

## Description

Return a table of common names

## Usage

```
common_names(
  species_list = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  Language = "English",
  fields = NULL
)
```

## Arguments

species_list	A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.
server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
version	a version string for the database, will default to the latest release. see [get_releases()] for details.
db	the
Language	a string specifying the language for the common name, e.g. "English"
fields	a character vector specifying which fields (columns) should be returned. By default, all available columns recognized by the parser are returned. Mostly for backwards compatibility as users can subset by column later

## Details

Note that there are many common names for a given sci name, so sci\_to\_common doesn't make sense

## Value

a data.frame of common names by species queried. If multiple species are queried, The resulting data.frames are concatenated.

---

common_to_sci	<i>common_to_sci</i>
---------------	----------------------

---

### Description

Return a list of scientific names corresponding to given the common name(s).

### Usage

```
common_to_sci(
  x,
  Language = "English",
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db()
)
```

### Arguments

x	a common name or list of common names
Language	a string specifying the language for the common name, e.g. "English"
server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
version	a version string for the database, will default to the latest release. see [get_releases()] for details.
db	the

### Details

If more than one scientific name matches the common name (e.g. "trout"), the function will simply return a list of all matching scientific names. If given more than one common name, the resulting strings of matching scientific names are simply concatenated.

### Value

a character vector of scientific names

### See Also

[species\\_list](#), [synonyms](#)

### Examples

```
common_to_sci(c("Bicolor cleaner wrasse", "humphead parrotfish"), Language="English")
common_to_sci(c("Coho Salmon", "trout"))
```

---

country	<i>country</i>
---------	----------------

---

## Description

return a table of country for the requested species, as reported in FishBASE.org

## Usage

```
country(
  species_list = NULL,
  fields = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```

## Arguments

species_list	A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.
fields	a character vector specifying which fields (columns) should be returned. By default, all available columns recognized by the parser are returned. Mostly for backwards compatibility as users can subset by column later
server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
version	a version string for the database, will default to the latest release. see [get_releases()] for details.
db	the
...	unused; for backwards compatibility only

## Details

e.g. <http://www.fishbase.us/Country>

## Examples

```
## Not run:
country(species_list(Genus='Labroides'))

## End(Not run)
```

---

 countrysub

*countrysub*


---

## Description

return a table of countrysub for the requested species

## Usage

```
countrysub(
  species_list = NULL,
  fields = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```

## Arguments

species_list	A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.
fields	a character vector specifying which fields (columns) should be returned. By default, all available columns recognized by the parser are returned. Mostly for backwards compatibility as users can subset by column later
server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
version	a version string for the database, will default to the latest release. see [get_releases()] for details.
db	the
...	unused; for backwards compatibility only

## Examples

```
## Not run:
countrysub(species_list(Genus='Labroides'))

## End(Not run)
```



countrysubref                      *countrysubref*

**Description**

return a table of countrysubref

**Usage**

```
countrysubref(
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```

**Arguments**

- server                      can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE\_API', e.g. 'Sys.setenv(FISHBASE\_API="sealifebase")'.
- version                    a version string for the database, will default to the latest release. see [get\_releases()] for details.
- db                              the
- ...                            unused; for backwards compatibility only

**Examples**

```
## Not run:
countrysubref()

## End(Not run)
```

c\_code                              *c\_code*

**Description**

return a table of country information for the requested c\_code, as reported in FishBASE.org

**Usage**

```

c_code(
  c_code = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)

```

**Arguments**

c_code	a C_Code or list of C_Codes (FishBase country code)
server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
version	a version string for the database, will default to the latest release. see [get_releases()] for details.
db	the
...	unused; for backwards compatibility only

**Details**

e.g. <http://www.fishbase.us/Country>

**Examples**

```

## Not run:
c_code(440)

## End(Not run)

```

---

default\_db

*Connect to the rfishbase database*

---

**Description**

Connect to the rfishbase database

**Usage**

```

default_db(dbdir = db_dir(), driver = Sys.getenv("DB_DRIVER", "RSQLite"))

```

**Arguments**

<code>dbdir</code>	Path to the database.
<code>driver</code>	Default driver, one of "duckdb", "MonetDBLite", "RSQLite". 'rfishbase' will select the first one of those it finds available if a driver is not set. This fallback can be overwritten either by explicit argument or by setting the environmental variable 'rfishbase_DRIVER'.

**Details**

This function provides a default database connection for 'rfishbase'. Note that you can use 'rfishbase' with any DBI-compatible database connection by passing the connection object directly to 'rfishbase' functions using the 'db' argument. 'default\_db()' exists only to provide reasonable automatic defaults based on what is available on your system.

'duckdb' or 'MonetDBLite' will give the best performance, and regular users 'rfishbase' will work with the built-in 'RSQLite', and with other database connections such as Postgres or MariaDB, but queries (filtering joins) will be much slower on these non-columnar databases.

For performance reasons, this function will also cache and restore the existing database connection, making repeated calls to 'default\_db()' much faster and more failsafe than repeated calls to [DBI::dbConnect]

**Value**

Returns a 'src\_dbi' connection to the default duckdb database

**Examples**

```
## OPTIONAL: you can first set an alternative home location,
## such as a temporary directory:
Sys.setenv(FISHBASE_HOME=tempdir())

## Connect to the database:
db <- default_db()
```

---

 diet

*diet*


---

**Description**

diet

**Usage**

```
diet(
  species_list = NULL,
  fields = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```

**Arguments**

<code>species_list</code>	A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.
<code>fields</code>	a character vector specifying which fields (columns) should be returned. By default, all available columns recognized by the parser are returned. Mostly for backwards compatibility as users can subset by column later
<code>server</code>	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
<code>version</code>	a version string for the database, will default to the latest release. see [get_releases()] for details.
<code>db</code>	the
<code>...</code>	unused; for backwards compatibility only

**Value**

a table of species diet

**References**

[http://www.fishbase.org/manual/english/fishbasethe\\_diet\\_table.htm](http://www.fishbase.org/manual/english/fishbasethe_diet_table.htm)

**Examples**

```
## Not run:
diet("Oreochromis niloticus")

## End(Not run)
```

---

diet_items	<i>fooditems</i>
------------	------------------

---

## Description

fooditems

## Usage

```
diet_items(
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```

## Arguments

server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
version	a version string for the database, will default to the latest release. see [get_releases()] for details.
db	the
...	unused; for backwards compatibility only

## Value

a table of species diet\_items

## Examples

```
## Not run:
diet_items("Oreochromis niloticus")

## End(Not run)
```

---

distribution	<i>distribution</i>
--------------	---------------------

---

### Description

return a table of species locations as reported in FishBASE.org FAO location data

### Usage

```
distribution(
  species_list = NULL,
  fields = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```

### Arguments

species_list	A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.
fields	a character vector specifying which fields (columns) should be returned. By default, all available columns recognized by the parser are returned. Mostly for backwards compatibility as users can subset by column later
server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
version	a version string for the database, will default to the latest release. see [get_releases()] for details.
db	the
...	unused; for backwards compatibility only

### Details

currently this is ~ FAO areas table (minus "note" field) e.g. <http://www.fishbase.us/Country/FaoAreaList.php?ID=5537>

### Examples

```
## Not run:
distribution(species_list(Genus='Labroides'))

## End(Not run)
```

---

docs

*docs*


---

**Description**

documentation of tables and fields

**Usage**

```
docs(table = NULL, server = NULL, ...)
```

**Arguments**

table	the table for which the documentation should be displayed. If no table is given, documentation summarizing all available tables is shown.
server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
...	unused; for backwards compatibility only

**Value**

A data.frame which lists the name of each table (if no table argument is given), along with a description of the table and a URL linking to further information about the table. If a specific table is named in the table argument, then the function will return a data.frame listing all the fields (columns) found in that table, a description of what the field label means, and the units in which the field is measured. These descriptions of the columns are not made available by FishBase and must be manually generated and curated by FishBase users. At this time, many fields are still missing. Please take a moment to fill in any fields you use in the source table here: <https://github.com/ropensci/fishbaseapi/tree/master/docs/docs-sources>

**Examples**

```
tables <- docs()
# Describe the fecundity table
dplyr::filter(tables, table == "fecundity")$description
## See fields in fecundity table
docs("fecundity")
## Note: only
```

---

 ecology

*ecology*


---

**Description**

ecology

**Usage**

```
ecology(
  species_list = NULL,
  fields = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```

**Arguments**

<code>species_list</code>	A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.
<code>fields</code>	a character vector specifying which fields (columns) should be returned. By default, all available columns recognized by the parser are returned. Mostly for backwards compatibility as users can subset by column later
<code>server</code>	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
<code>version</code>	a version string for the database, will default to the latest release. see [get_releases()] for details.
<code>db</code>	the
<code>...</code>	unused; for backwards compatibility only

**Details**

By default, will only return one entry (row) per species. Increase limit to get multiple returns for different stocks of the same species, though often data is either identical to the first or simply missing in the additional stocks.

**Value**

a table of species ecology data

**References**

[http://www.fishbase.org/manual/english/fishbasethe\\_ecology\\_table.htm](http://www.fishbase.org/manual/english/fishbasethe_ecology_table.htm)



**Examples**

```
## Not run:
ecology("Oreochromis niloticus")

## trophic levels and standard errors for a list of species
ecology(c("Oreochromis niloticus", "Salmo trutta"),
        fields=c("SpecCode", "FoodTroph", "FoodSeTroph", "DietTroph", "DietSeTroph"))

## End(Not run)
```

---

ecosystem	<i>ecosystem</i>
-----------	------------------

---

**Description**

ecosystem

**Usage**

```
ecosystem(
  species_list = NULL,
  fields = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```

**Arguments**

<code>species_list</code>	A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.
<code>fields</code>	a character vector specifying which fields (columns) should be returned. By default, all available columns recognized by the parser are returned. Mostly for backwards compatibility as users can subset by column later
<code>server</code>	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
<code>version</code>	a version string for the database, will default to the latest release. see [get_releases()] for details.
<code>db</code>	the
<code>...</code>	unused; for backwards compatibility only

**Value**

a table of species ecosystems data

**Examples**

```
## Not run:
ecosystem("Oreochromis niloticus")

## End(Not run)
```

---

 estimate
 

---



---

*estimate*


---

**Description**

estimate

**Usage**

```
estimate(
  species_list = NULL,
  fields = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```

**Arguments**

species_list	A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.
fields	a character vector specifying which fields (columns) should be returned. By default, all available columns recognized by the parser are returned. Mostly for backwards compatibility as users can subset by column later
server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
version	a version string for the database, will default to the latest release. see [get_releases()] for details.
db	the
...	unused; for backwards compatibility only

**Value**

a table of estimates from some models on trophic levels

**References**

[http://www.fishbase.us/manual/English/FishbaseThe\\_FOOD\\_ITEMS\\_table.htm](http://www.fishbase.us/manual/English/FishbaseThe_FOOD_ITEMS_table.htm)

**Examples**

```
## Not run:
estimate("Oreochromis niloticus")

## End(Not run)
```

---

faoareas	<i>faoareas</i>
----------	-----------------

---

**Description**

return a table of species locations as reported in FishBASE.org FAO location data

**Usage**

```
faoareas(
  species_list = NULL,
  fields = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```

**Arguments**

<code>species_list</code>	A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.
<code>fields</code>	a character vector specifying which fields (columns) should be returned. By default, all available columns recognized by the parser are returned. Mostly for backwards compatibility as users can subset by column later
<code>server</code>	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
<code>version</code>	a version string for the database, will default to the latest release. see [get_releases()] for details.
<code>db</code>	the
<code>...</code>	unused; for backwards compatibility only

**Details**

currently this is ~ FAO areas table (minus "note" field) e.g. <http://www.fishbase.us/Country/FaoAreaList.php?ID=5537>

**Value**

a tibble, empty tibble if no results found

**Examples**

```
## Not run:
  faoareas()

## End(Not run)
```

---

 fecundity
 

---



---

*fecundity*


---

**Description**

fecundity

**Usage**

```
fecundity(
  species_list = NULL,
  fields = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```

**Arguments**

species_list	A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.
fields	a character vector specifying which fields (columns) should be returned. By default, all available columns recognized by the parser are returned. Mostly for backwards compatibility as users can subset by column later
server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
version	a version string for the database, will default to the latest release. see [get_releases()] for details.
db	the
...	unused; for backwards compatibility only

**Value**

a table of species fecundity

**Examples**

```
## Not run:
fecundity("Oreochromis niloticus")

## End(Not run)
```

---

fishbase	<i>A table of all the the species found in FishBase, including taxonomic classification and the Species Code (SpecCode) by which the species is identified in FishBase.</i>
----------	---

---

**Description**

A table of all the the species found in FishBase, including taxonomic classification and the Species Code (SpecCode) by which the species is identified in FishBase.

**Author(s)**

Carl Boettiger <carl@ropensci.org>

---

fooditems	<i>fooditems</i>
-----------	------------------

---

**Description**

fooditems

**Usage**

```
fooditems(
  species_list = NULL,
  fields = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```

**Arguments**

species_list	A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.
fields	a character vector specifying which fields (columns) should be returned. By default, all available columns recognized by the parser are returned. Mostly for backwards compatibility as users can subset by column later
server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
version	a version string for the database, will default to the latest release. see [get_releases()] for details.
db	the
...	unused; for backwards compatibility only

**Value**

a table of species fooditems

**References**

[http://www.fishbase.org/manual/english/fishbasethe\\_food\\_items\\_table.htm](http://www.fishbase.org/manual/english/fishbasethe_food_items_table.htm)

**Examples**

```
## Not run:
fooditems("Oreochromis niloticus")

## End(Not run)
```

---

genetics

*genetics*

---

**Description**

genetics

**Usage**

```
genetics(
  species_list = NULL,
  fields = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```

**Arguments**

species_list	A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.
fields	a character vector specifying which fields (columns) should be returned. By default, all available columns recognized by the parser are returned. Mostly for backwards compatibility as users can subset by column later
server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
version	a version string for the database, will default to the latest release. see [get_releases()] for details.
db	the
...	unused; for backwards compatibility only

**Value**

a table of species genetics data

**Examples**

```
## Not run:
genetics("Oreochromis niloticus")
genetics("Labroides dimidiatus")

## End(Not run)
```

---

introductions

*introductions*


---

**Description**

introductions

**Usage**

```
introductions(
  species_list = NULL,
  fields = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```

**Arguments**

species_list	A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.
fields	a character vector specifying which fields (columns) should be returned. By default, all available columns recognized by the parser are returned. Mostly for backwards compatibility as users can subset by column later
server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
version	a version string for the database, will default to the latest release. see [get_releases()] for details.
db	the
...	unused; for backwards compatibility only

**Value**

a table of species introductions data

**Examples**

```
## Not run:
introductions("Oreochromis niloticus")

## End(Not run)
```

---

larvae

*larvae*

---

**Description**

larvae

**Usage**

```
larvae(
  species_list = NULL,
  fields = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```



**Arguments**

species_list	A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.
fields	a character vector specifying which fields (columns) should be returned. By default, all available columns recognized by the parser are returned. Mostly for backwards compatibility as users can subset by column later
server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
version	a version string for the database, will default to the latest release. see [get_releases()] for details.
db	the
...	unused; for backwards compatibility only

**Value**

a table of larval data

**Examples**

```
## Not run:
larvae("Oreochromis niloticus")

## End(Not run)
```

---

length_freq	<i>length_freq</i>
-------------	--------------------

---

**Description**

return a table of species fooditems

**Usage**

```
length_freq(
  species_list = NULL,
  fields = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```

**Arguments**

species_list	A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.
fields	a character vector specifying which fields (columns) should be returned. By default, all available columns recognized by the parser are returned. Mostly for backwards compatibility as users can subset by column later
server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
version	a version string for the database, will default to the latest release. see [get_releases()] for details.
db	the
...	unused; for backwards compatibility only

**Value**

a table of length\_freq information by species; see details

**References**

<http://www.fishbase.org/manual/english/lengthfrequency.htm>

**Examples**

```
## Not run:
length_freq("Oreochromis niloticus")

## End(Not run)
```

---

length_length	<i>length_length</i>
---------------	----------------------

---

**Description**

return a table of lengths

**Usage**

```
length_length(
  species_list = NULL,
  fields = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```

**Arguments**

species_list	A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.
fields	a character vector specifying which fields (columns) should be returned. By default, all available columns recognized by the parser are returned. Mostly for backwards compatibility as users can subset by column later
server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
version	a version string for the database, will default to the latest release. see [get_releases()] for details.
db	the
...	unused; for backwards compatibility only

**Details**

This table contains relationships for the conversion of one length type to another for over 8,000 species of fish, derived from different publications, e.g. Moutopoulos and Stergiou (2002) and Gaygusuz et al (2006), or from fish pictures, e.g. Collette and Nauen (1983), Compagno (1984) and Randall (1997). The relationships, which always refer to centimeters, may consist either of a regression linking two length types, of the form: Length type (2) = a + b x Length type (1) Length type (2) = b' x Length type (1) The available length types are, as elsewhere in FishBase, TL = total length; FL = fork length; SL = standard length; WD = width (in rays); OT = other type (to be specified in the Comment field). When a version of equation (1) is presented, the length range, the number of fish used in the regression, the sex and the correlation coefficient are presented, if available. When a version of equation (2) is presented, the range and the correlation coefficient are omitted, as the ratio in (2) will usually be estimated from a single specimen, or a few fish covering a narrow range of lengths.

**Value**

a table of lengths

**References**

[http://www.fishbase.org/manual/english/PDF/FB\\_Book\\_CBinohlan\\_Length-Length\\_RF\\_JG.pdf](http://www.fishbase.org/manual/english/PDF/FB_Book_CBinohlan_Length-Length_RF_JG.pdf)

**Examples**

```
## Not run:
length_length("Oreochromis niloticus")

## End(Not run)
```

---

length_weight	<i>length_weight</i>
---------------	----------------------

---

### Description

The LENGTH-WEIGHT table presents the a and b values of over 5,000 length-weight relationships of the form  $W = a \times L^b$ , pertaining to about over 2,000 fish species.

### Usage

```
length_weight(
  species_list = NULL,
  fields = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```

### Arguments

species_list	A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.
fields	a character vector specifying which fields (columns) should be returned. By default, all available columns recognized by the parser are returned. Mostly for backwards compatibility as users can subset by column later
server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
version	a version string for the database, will default to the latest release. see [get_releases()] for details.
db	the
...	unused; for backwards compatibility only

### Details

See references for official documentation. From FishBase.org: Length-weight relationships are important in fisheries science, notably to raise length-frequency samples to total catch, or to estimate biomass from underwater length observations. The units of length and weight in FishBase are centimeter and gram, respectively. Thus when length-weight relationships are not in cm-g, the intercept 'a' is transformed as follows:

$$a'(cm, g) = a(mm, g) \cdot 10^b \quad a'(cm, g) = a(cm, kg) \cdot 1000 \quad a'(cm, g) = a(mm, mg) \cdot 10^b / 1000 \quad a'(cm, g) = a(mm, kg) \cdot 10^b \cdot 1000$$

However, published length-weight relationships are sometimes difficult to use, as they may be based on a length measurement type (e.g., fork length) different from ones length measurements (expressed e.g., as total length). Therefore, to facilitate conversion between length types, an additional

LENGTH-LENGTH table, # presented below, was devised which presents linear regressions or ratios linking length types (e.g., FL vs. TL). We included a calculated field with the weight of a 10 cm fish (which should be in the order of 10 g for normal, fusiform shaped fish), to allow identification of gross errors, given knowledge of the body form of a species.

### Value

a table of length\_weight information by species; see details

### References

[http://www.fishbase.org/manual/english/fishbasethe\\_length\\_weight\\_table.htm](http://www.fishbase.org/manual/english/fishbasethe_length_weight_table.htm)

### Examples

```
## Not run:
length_weight("Oreochromis niloticus")

## End(Not run)
```

---

list_fields	<i>list_fields</i>
-------------	--------------------

---

### Description

list fields

### Usage

```
list_fields(fields = NULL, server = NULL, implemented_only = TRUE)
```

### Arguments

**fields** field (column name) to search for

**server** base URL to the FishBase API (by default). For SeaLifeBase, use <https://fishbase.ropensci.org/sealifebase>

**implemented\_only** by default, only return those tables that have been implemented.

### Details

Calling 'list\_fields()' with no arguments will return the full table of all known fields. Then users can employ standard filter techniques like `grep` for partial name matching; see examples.

### Value

a data frame listing the table names (matching function names in `rfishbase`) and the matching column names those tables have implemented.

**Examples**

```
list_fields("Temp")

## Regex matching on full table
library(dplyr)
list_fields() %>% filter(grepl("length", columns, ignore.case = TRUE))
```

---

load\_taxa

*load\_taxa*


---

**Description**

load\_taxa

**Usage**

```
load_taxa(
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  collect = TRUE,
  ...
)
```

**Arguments**

server	Either "fishbase" (the default) or "sealifebase"
version	the version of the database you want. Will default to the latest available; see [available_releases()].
db	A remote database connection. Will default to the best available system, see [default_db()].
collect	return a data.frame if TRUE, otherwise, a DBI connection to the table in the database
...	for compatibility with previous versions

**Value**

the taxa list

---

maturity	<i>maturity</i>
----------	-----------------

---

**Description**

maturity

**Usage**

```
maturity(
  species_list = NULL,
  fields = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```

**Arguments**

species_list	A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.
fields	a character vector specifying which fields (columns) should be returned. By default, all available columns recognized by the parser are returned. Mostly for backwards compatibility as users can subset by column later
server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
version	a version string for the database, will default to the latest release. see [get_releases()] for details.
db	the
...	unused; for backwards compatibility only

**Value**

a table of species maturity

**Examples**

```
## Not run:
maturity("Oreochromis niloticus")

## End(Not run)
```

---

 morphology

*morphology*


---

### Description

morphology

### Usage

```
morphology(
  species_list = NULL,
  fields = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```

### Arguments

species_list	A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.
fields	a character vector specifying which fields (columns) should be returned. By default, all available columns recognized by the parser are returned. Mostly for backwards compatibility as users can subset by column later
server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
version	a version string for the database, will default to the latest release. see [get_releases()] for details.
db	the
...	unused; for backwards compatibility only

### Value

a table of species morphology data

### Examples

```
## Not run:
morphology("Oreochromis niloticus")

## End(Not run)
```



---

morphometrics	<i>morphometrics</i>
---------------	----------------------

---

**Description**

morphometrics

**Usage**

```
morphometrics(
  species_list = NULL,
  fields = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```

**Arguments**

species_list	A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.
fields	a character vector specifying which fields (columns) should be returned. By default, all available columns recognized by the parser are returned. Mostly for backwards compatibility as users can subset by column later
server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
version	a version string for the database, will default to the latest release. see [get_releases()] for details.
db	the
...	unused; for backwards compatibility only

**Value**

a table of species morphometrics data

**Examples**

```
## Not run:
morphometrics("Oreochromis niloticus")

## End(Not run)
```

---

occurrence	<i>occurrence</i>
------------	-------------------

---

**Description**

occurrence

**Usage**

occurrence()

**Details**

THE OCCURRENCE TABLE HAS BEEN DROPPED BY FISHBASE - THIS FUNCTION NOW RETURNS A STOP MESSAGE.

---

oxygen	<i>oxygen</i>
--------	---------------

---

**Description**

oxygen

**Usage**

```
oxygen(
  species_list = NULL,
  fields = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```

**Arguments**

species_list	A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.
fields	a character vector specifying which fields (columns) should be returned. By default, all available columns recognized by the parser are returned. Mostly for backwards compatibility as users can subset by column later
server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.

version	a version string for the database, will default to the latest release. see [get_releases()] for details.
db	the
...	unused; for backwards compatibility only

**Value**

a table of species oxygen data

**Examples**

```
## Not run:
oxygen("Oreochromis niloticus")

## End(Not run)
```

---

popchar	<i>popchar</i>
---------	----------------

---

**Description**

Table of maximum length (Lmax), weight (Wmax) and age (tmax)

**Usage**

```
popchar(
  species_list = NULL,
  fields = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```

**Arguments**

species_list	A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.
fields	a character vector specifying which fields (columns) should be returned. By default, all available columns recognized by the parser are returned. Mostly for backwards compatibility as users can subset by column later
server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
version	a version string for the database, will default to the latest release. see [get_releases()] for details.
db	the
...	unused; for backwards compatibility only

**Details**

See references for official documentation. From FishBase.org: This table presents information on maximum length (Lmax), weight (Wmax) and age (tmax) from various localities where a species occurs. The largest values from this table are also entered in the SPECIES table. The POPCHAR table also indicates whether the Lmax, Wmax and tmax values or various combinations thereof refer to the same individual fish.

**References**

[http://www.fishbase.org/manual/english/fishbasethe\\_popchar\\_table.htm](http://www.fishbase.org/manual/english/fishbasethe_popchar_table.htm)

**Examples**

```
## Not run:
popchar("Oreochromis niloticus")

## End(Not run)
```

---

popgrowth

*popgrowth*

---

**Description**

This table contains information on growth, natural mortality and length at first maturity, which serve as inputs to many fish stock assessment models. The data can also be used to generate empirical relationships between growth parameters or natural mortality estimates, and their correlates (e.g., body shape, temperature, etc.), a line of research that is useful both for stock assessment and for increasing understanding of the evolution of life-history strategies

**Usage**

```
popgrowth(
  species_list = NULL,
  fields = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```

**Arguments**

**species\_list** A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.

**fields** a character vector specifying which fields (columns) should be returned. By default, all available columns recognized by the parser are returned. Mostly for backwards compatibility as users can subset by column later

server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
version	a version string for the database, will default to the latest release. see [get_releases()] for details.
db	the
...	unused; for backwards compatibility only

**Value**

a table of population growth information by species; see details

**References**

[http://www.fishbase.org/manual/english/fishbasethe\\_popgrowth\\_table.htm](http://www.fishbase.org/manual/english/fishbasethe_popgrowth_table.htm)

**Examples**

```
## Not run:
popgrowth("Oreochromis niloticus")

## End(Not run)
```

---

popqb

*popqb*

---

**Description**

popqb

**Usage**

```
popqb(
  species_list = NULL,
  fields = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```

**Arguments**

species_list	A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.
fields	a character vector specifying which fields (columns) should be returned. By default, all available columns recognized by the parser are returned. Mostly for backwards compatibility as users can subset by column later
server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
version	a version string for the database, will default to the latest release. see [get_releases()] for details.
db	the
...	unused; for backwards compatibility only

**Value**

a table of species popqb

**References**

[http://www.fishbase.org/manual/english/fishbasethe\\_popqb\\_table.htm](http://www.fishbase.org/manual/english/fishbasethe_popqb_table.htm)

**Examples**

```
## Not run:
popqb("Oreochromis niloticus")

## End(Not run)
```

---

predators

*predators*

---

**Description**

predators

**Usage**

```
predators(
  species_list = NULL,
  fields = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```

**Arguments**

species_list	A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.
fields	a character vector specifying which fields (columns) should be returned. By default, all available columns recognized by the parser are returned. Mostly for backwards compatibility as users can subset by column later
server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
version	a version string for the database, will default to the latest release. see [get_releases()] for details.
db	the
...	unused; for backwards compatibility only

**Value**

a table of predators

**References**

[http://www.fishbase.org/manual/english/fishbasethe\\_predators\\_table.htm](http://www.fishbase.org/manual/english/fishbasethe_predators_table.htm)

**Examples**

```
## Not run:
predators("Oreochromis niloticus")

## End(Not run)
```

---

ration	<i>ration</i>
--------	---------------

---

**Description**

ration

**Usage**

```
ration(
  species_list = NULL,
  fields = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```

**Arguments**

species_list	A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.
fields	a character vector specifying which fields (columns) should be returned. By default, all available columns recognized by the parser are returned. Mostly for backwards compatibility as users can subset by column later
server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
version	a version string for the database, will default to the latest release. see [get_releases()] for details.
db	the
...	unused; for backwards compatibility only

**Value**

a table of species ration

**References**

[http://www.fishbase.org/manual/english/fishbasethe\\_ration\\_table.htm](http://www.fishbase.org/manual/english/fishbasethe_ration_table.htm)

**Examples**

```
## Not run:
ration("Oreochromis niloticus")

## End(Not run)
```

---

references

*references*

---

**Description**

references

**Usage**

```
references(
  codes = NULL,
  fields = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```



**Arguments**

codes	One or more Fishbase reference numbers, matching the RefNo field
fields	a character vector specifying which fields (columns) should be returned. By default, all available columns recognized by the parser are returned. Mostly for backwards compatibility as users can subset by column later
server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
version	a version string for the database, will default to the latest release. see [get_releases()] for details.
db	the
...	unused; for backwards compatibility only

**Value**

a tibble (data.frame) of reference data

**Examples**

```
## Not run:
references(codes = 1)
references(codes = 1:6)
references(codes = 1:6, fields = c('Author', 'Year', 'Title'))
references() # all references

## End(Not run)
```

---

reproduction	<i>reproduction</i>
--------------	---------------------

---

**Description**

reproduction

**Usage**

```
reproduction(
  species_list = NULL,
  fields = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```

**Arguments**

species_list	A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.
fields	a character vector specifying which fields (columns) should be returned. By default, all available columns recognized by the parser are returned. Mostly for backwards compatibility as users can subset by column later
server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
version	a version string for the database, will default to the latest release. see [get_releases()] for details.
db	the
...	unused; for backwards compatibility only

**Value**

a table of species reproduction

**Examples**

```
## Not run:
reproduction("Oreochromis niloticus")

## End(Not run)
```

---

sealifebase	<i>A table of all the the species found in SeaLifeBase, including taxonomic classification and the Species Code (SpecCode) by which the species is identified in SeaLifeBase</i>
-------------	--

---

**Description**

A table of all the the species found in SeaLifeBase, including taxonomic classification and the Species Code (SpecCode) by which the species is identified in SeaLifeBase

**Author(s)**

Carl Boettiger <carl@ropensci.org>

---

spawning	<i>spawning</i>
----------	-----------------

---

**Description**

spawning

**Usage**

```
spawning(
  species_list = NULL,
  fields = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```

**Arguments**

<code>species_list</code>	A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.
<code>fields</code>	a character vector specifying which fields (columns) should be returned. By default, all available columns recognized by the parser are returned. Mostly for backwards compatibility as users can subset by column later
<code>server</code>	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
<code>version</code>	a version string for the database, will default to the latest release. see [get_releases()] for details.
<code>db</code>	the
<code>...</code>	unused; for backwards compatibility only

**Value**

a table of species spawning

**Examples**

```
## Not run:
spawning("Oreochromis niloticus")

## End(Not run)
```

---

species	<i>species</i>
---------	----------------

---

### Description

Provide wrapper to work with species lists.

### Usage

```
species(
  species_list = NULL,
  fields = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```

### Arguments

species_list	A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.
fields	a character vector specifying which fields (columns) should be returned. By default, all available columns recognized by the parser are returned. Mostly for backwards compatibility as users can subset by column later
server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
version	a version string for the database, will default to the latest release. see [get_releases()] for details.
db	the
...	unused; for backwards compatibility only

### Details

The Species table is the heart of FishBase. This function provides a convenient way to query, tidy, and assemble data from that table given an entire list of species. For details, see: <http://www.fishbase.org/manual/english/fishb>

Species scientific names are defined according to fishbase taxonomy and nomenclature.

### Value

a data.frame with rows for species and columns for the fields returned by the query (FishBase 'species' table)

**Examples**

```
## Not run:

species(c("Labroides bicolor", "Bolbometopon muricatum"))
species(c("Labroides bicolor", "Bolbometopon muricatum"), fields = species_fields$habitat)

## End(Not run)
```

---

species\_by\_ecosystem *Species list by ecosystem*

---

**Description**

Species list by ecosystem

**Usage**

```
species_by_ecosystem(
  ecosystem,
  species_list = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```

**Arguments**

ecosystem	(character) an ecosystem name
species_list	A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.
server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
version	a version string for the database, will default to the latest release. see [get_releases()] for details.
db	the
...	unused; for backwards compatibility only

**Value**

a table of species ecosystems data

**Examples**

```
## Not run:  
species_by_ecosystem(ecosystem = "Arctic", server = "sealifebase")  
  
## End(Not run)
```

---

species_fields	<i>A list of the species_fields available</i>
----------------	---

---

**Description**

A list of the species\_fields available

**Author(s)**

Carl Boettiger <carl@ropensci.org>

---

species_list	<i>species_list</i>
--------------	---------------------

---

**Description**

Return the a species list given a taxonomic group

**Usage**

```
species_list(  
  Class = NULL,  
  Order = NULL,  
  Family = NULL,  
  Subfamily = NULL,  
  Genus = NULL,  
  Species = NULL,  
  SpecCode = NULL,  
  SuperClass = NULL,  
  server = getOption("FISHBASE_API", FISHBASE_API)  
)
```

**Arguments**

Class	Request all species in this taxonomic Class
Order	Request all species in this taxonomic Order
Family	Request all species in this taxonomic Family
Subfamily	Request all species in this taxonomic SubFamily
Genus	Request all species in this taxonomic Genus
Species	Request all species in this taxonomic Species
SpecCode	Request species name of species matching this SpecCode
SuperClass	Request all species of this Superclass
server	fishbase or sealifebase

**Examples**

```
## All species in the Family
species_list(Family = 'Scaridae')
## All species in the Genus
species_list(Genus = 'Labroides')
```

---

species_names	<i>species names</i>
---------------	----------------------

---

**Description**

returns species names given FishBase's SpecCodes

**Usage**

```
species_names(
  codes,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db()
)
```

**Arguments**

codes	a vector of speccodes (e.g. column from a table)
server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
version	a version string for the database, will default to the latest release. see [get_releases()] for details.
db	the

**Value**

A character vector of species names for the SpecCodes

---

speed	<i>speed</i>
-------	--------------

---

**Description**

speed

**Usage**

```
speed(
  species_list = NULL,
  fields = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```

**Arguments**

species_list	A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.
fields	a character vector specifying which fields (columns) should be returned. By default, all available columns recognized by the parser are returned. Mostly for backwards compatibility as users can subset by column later
server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
version	a version string for the database, will default to the latest release. see [get_releases()] for details.
db	the
...	unused; for backwards compatibility only

**Value**

a table of species speed data

**Examples**

```
## Not run:
speed("Oreochromis niloticus")

## End(Not run)
```



---

stocks	<i>stocks</i>
--------	---------------

---

## Description

stocks

## Usage

```
stocks(
  species_list = NULL,
  fields = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```

## Arguments

<code>species_list</code>	A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.
<code>fields</code>	a character vector specifying which fields (columns) should be returned. By default, all available columns recognized by the parser are returned. Mostly for backwards compatibility as users can subset by column later
<code>server</code>	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
<code>version</code>	a version string for the database, will default to the latest release. see [get_releases()] for details.
<code>db</code>	the
<code>...</code>	unused; for backwards compatibility only

## Value

a table of species stocks data

## Examples

```
## Not run:
stocks("Oreochromis niloticus")

## End(Not run)
```

---

swimming	<i>swimming</i>
----------	-----------------

---

## Description

swimming

## Usage

```
swimming(
  species_list = NULL,
  fields = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```

## Arguments

species_list	A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.
fields	a character vector specifying which fields (columns) should be returned. By default, all available columns recognized by the parser are returned. Mostly for backwards compatibility as users can subset by column later
server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
version	a version string for the database, will default to the latest release. see [get_releases()] for details.
db	the
...	unused; for backwards compatibility only

## Value

a table of species swimming data

## Examples

```
## Not run:
swimming("Oreochromis niloticus")

## End(Not run)
```

---

 synonyms

---

*synonyms*


---

**Description**

Check for alternate versions of a scientific name

**Usage**

```
synonyms(
  species_list = NULL,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```

**Arguments**

species_list	A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.
server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
version	a version string for the database, will default to the latest release. see [get_releases()] for details.
db	the
...	unused; for backwards compatibility only

**Details**

For further information on fields returned, see: [http://www.fishbase.org/manual/english/fishbasethe\\_synonyms\\_table.htm](http://www.fishbase.org/manual/english/fishbasethe_synonyms_table.htm)

**Value**

A table with information about the synonym. Will generally be only a single row if a species name is given. If a FishBase SpecCode is given, all synonyms matching that SpecCode are shown, and the table indicates which one is Valid for FishBase. This may or may not match the valid name for Catalog of Life (Col), also shown in the table. See examples for details.

---

validate_names	<i>validate_names</i>
----------------	-----------------------

---

**Description**

Check for alternate versions of a scientific name and return the scientific names FishBase recognizes as valid

**Usage**

```
validate_names(
  species_list,
  server = getOption("FISHBASE_API", "fishbase"),
  version = get_latest_release(),
  db = default_db(),
  ...
)
```

**Arguments**

species_list	A vector of scientific names (each element as "genus species"). If empty, a table for all fish will be returned.
server	can be set to either "fishbase" or "sealifebase" to switch between databases. NOTE: it is usually easier to leave this as NULL and set the source instead using the environmental variable 'FISHBASE_API', e.g. 'Sys.setenv(FISHBASE_API="sealifebase")'.
version	a version string for the database, will default to the latest release. see [get_releases()] for details.
db	the
...	unused; for backwards compatibility only

**Value**

a string of the validated names

**Examples**

```
validate_names("Abramites ternetzi")
```

# Index

- \* **data**
  - fishbase, 21
  - sealifebase, 42
  - species\_fields, 46
- \* **package**
  - rfishbase-package, 3
- available\_releases, 3
- brains, 4
- c\_code, 9
- common\_names, 5
- common\_to\_sci, 6
- country, 7
- countrysub, 8
- countrysubref, 9
- default\_db, 10
- diet, 11
- diet\_items, 13
- distribution, 14
- docs, 15
- ecology, 16
- ecosystem, 17
- estimate, 18
- faoareas, 19
- fecundity, 20
- fishbase, 21
- fooditems, 21
- genetics, 22
- introductions, 23
- larvae, 24
- length\_freq, 25
- length\_length, 26
- length\_weight, 28
- list\_fields, 29
- load\_taxa, 30
- maturity, 31
- morphology, 32
- morphometrics, 33
- occurrence, 34
- oxygen, 34
- popchar, 35
- popgrowth, 36
- poplf (length\_freq), 25
- popll (length\_length), 26
- poplw (length\_weight), 28
- popqb, 37
- predators, 38
- ration, 39
- references, 40
- reproduction, 41
- rfishbase (rfishbase-package), 3
- rfishbase-package, 3
- sci\_to\_common (common\_names), 5
- sealifebase, 42
- spawning, 43
- species, 44
- species\_by\_ecosystem, 45
- species\_fields, 46
- species\_info (species), 44
- species\_list, 6, 46
- species\_names, 47
- speed, 48
- stocks, 49
- swimming, 50
- synonyms, 6, 51
- validate\_names, 52