Package: redlist (via r-universe)

October 13, 2026

Type Package

Title Interface to the IUCN Red List Data with Risk Metrics

Version 0.3.0

Description Access species conservation data from the International Union for Conservation of Nature (IUCN) Red List API <https://api.iucnredlist.org/api-docs/index.html>, including assessments, taxonomy, threats, habitats and historical status. The package also reconciles taxonomic names between the IUCN Red List and the Global Biodiversity Information Facility (GBIF), retrieves and checks GBIF occurrence records, and computes the range and population metrics of the IUCN Red List Categories and Criteria (IUCN Standards and Petitions Committee, 2024, <https://cmsdocs.s3.amazonaws.com/RedListGuidelines.pdf>): extent of occurrence and area of occupancy for criterion B, and population reduction for criterion A.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

Roxygen list(markdown = TRUE)

Imports cli (>= 3.6.3), dplyr (>= 1.1.4), httr2 (>= 1.1.2), magrittr (>= 2.0.3), methods, rlang (>= 1.1.0), rvest (>= 1.0.4), stats, utils

URL https://github.com/stangandaho/redlist, https://stangandaho.github.io/redlist/

BugReports https://github.com/stangandaho/redlist/issues

Depends R (>= 3.6.0)

Suggests CoordinateCleaner (>= 3.0.0), ggplot2 (>= 3.5.2), kableExtra (>= 1.4.0), knitr (>= 1.50), rgbif (>= 3.7.0), rmarkdown (>= 2.29), scales (>= 1.4.0), sf (>= 1.0.0), testthat (>= 3.0.0)

Config/testthat/edition 3

Config/Needs/website rmarkdown

VignetteBuilder knitr

Config/pak/sysreqs libxml2-dev libssl-dev

Repository https://stangandaho.r-universe.dev

Date/Publication 2026-10-13 22:54:05 UTC

RemoteUrl https://github.com/stangandaho/redlist

RemoteRef HEAD

RemoteSha a40a01b66520acdcdfc7c8dbde4e7028a603d352

Contents

rl_aoo . . . 3
rl_assessment_id . . . 4
rl_biogeographical_realms . . . 5
rl_check_api . . . 6
rl_check_occurrences . . . 7
rl_classes . . . 9
rl_comprehensive_groups . . . 10
rl_conservation_actions . . . 11
rl_countries . . . 13
rl_eoo . . . 14
rl_family . . . 15
rl_faos . . . 17
rl_generation_length . . . 18
rl_green_status . . . 20
rl_growth_forms . . . 21
rl_habitats . . . 22
rl_kingdoms . . . 23
rl_name_resolve . . . 25
rl_occurrences . . . 26
rl_open_file . . . 29
rl_orders . . . 29
rl_overall_reduction . . . 31
rl_phylum . . . 32
rl_population_trends . . . 33
rl_possibly_extinct . . . 34
rl_possibly_extinct_in_wild . . . 35
rl_red_list_categories . . . 36
rl_reduction . . . 37
rl_research . . . 39
rl_scientific_name . . . 40
rl_scopes . . . 41
rl_set_api . . . 43
rl_sis . . . 43

rl_statistics 44
rl_stresses 44
rl_systems 46
rl_threat_category 47
rl_threats 49
rl_use_and_trade 50
rl_version 51

Index **53**

`rl_aoo` *Area of occupancy (AOO) for IUCN criterion B*

Description

Compute the area of occupancy from occurrence records by counting the number of occupied grid cells and multiplying by the cell area, following equation 4.1 of the IUCN Red List guidelines (section 4.10). The guidelines require a reference scale of 2 by 2 km cells (an area of 4 square kilometres), which is the default here.

Usage

```
rl_aoo(x, coords = c("longitude", "latitude"), crs = 4326, cell_size = 2000)
```

Arguments

`x`	Occurrence records: an `sf` POINT object (for example the output of `rl_occurrences()`), or a data frame holding longitude and latitude columns.
`coords`	Character vector of length two giving the longitude and latitude column names when `x` is a data frame. Default `c("longitude", "latitude")`.
`crs`	Coordinate reference system of the input coordinates, passed to `sf`. Used only when `x` is a data frame or an `sf` object without a CRS. Default `4326` (WGS84).
`cell_size`	Grid cell side length in metres. Default `2000` (the IUCN reference scale of 2 by 2 km). Change it only with good reason, since the criterion B2 thresholds assume this scale.

Details

Geographic coordinates (longitude and latitude) are projected to a local Lambert azimuthal equal area system before the grid is applied, so cell sizes are measured in metres regardless of the input coordinate system.

The count uses a single grid whose origin is fixed at the projected origin. The guidelines note that shifting the grid can change the count and that the smallest estimate should then be used; that refinement is left to the assessor.

`sf` is required and is asked for interactively when it is not installed.

Value

A one row `sf` object whose `geometry` column holds the occupied grid cells as a MULTIPOLYGON, returned in the input coordinate system, and the columns:

`metric` `"AOO"`.

`area_km2` the area of occupancy in square kilometres.

`n_records` the number of records used.

`n_occupied_cells` the number of occupied grid cells.

`cell_size_km` the cell side length in kilometres.

`category_b2` the most threatened criterion B2 band the area reaches (`"CR"`, `"EN"` or `"VU"`), or `NA` when it reaches none. This is the spatial threshold only, not a full assessment, which also requires the criterion B subconditions.

References

IUCN Standards and Petitions Committee. 2024. Guidelines for Using the IUCN Red List Categories and Criteria. Version 16, section 4.10. https://cmsdocs.s3.amazonaws.com/RedListGuidelines.pdf

See Also

`rl_eoo()`

Examples

```
## Not run:
occ <- data.frame(
  longitude = c(2.1, 2.6, 3.0, 2.4, 2.9),
  latitude = c(9.1, 9.5, 9.0, 9.8, 9.3)
)
rl_aoo(occ)

## End(Not run)
```

`rl_assessment_id` *IUCN Red List assessment*

Description

Retrieves an assessment

Usage

```
rl_assessment_id(assessment_id = 1425064)
```

Arguments

`assessment_id` Assessment ID

Value

A tibble where each column represents a unique API response JSON key for the supplied `assessment_id`. The columns include key information about the Red List assessment, such as taxon details, category, year, and other relevant metadata.

Examples

```
## Not run:
rl_assessment_id(1425064)

## End(Not run)
```

`rl_biogeographical_realms` *IUCN Red List biogeographical realms*

Description

Retrieve available biogeographical realms or detailed species assessments for one or more realms.

Usage

```
rl_biogeographical_realms(
  code = NULL,
  year_published = NULL,
  latest = NULL,
  possibly_extinct = NULL,
  possibly_extinct_in_the_wild = NULL,
  scope_code = NULL,
  page = 1
)
```

Arguments

`code`	Numeric or Character. One or more biogeographical realm codes (e.g. `0` or `"0"`). Use `rl_biogeographical_realms()` to list available realms.
`year_published`	Optional. Single or numeric vector of years to filter assessments by publication year.
`latest`	Optional. Logical. If `TRUE`, return only the latest assessment per species.
`possibly_extinct`	Optional. Logical. Filter for species flagged as possibly extinct.
`possibly_extinct_in_the_wild`	Optional. Logical. Filter for species possibly extinct in the wild.
`scope_code`	Optional. Integer One or more scope codes to filter assessments.
`page`	Optional. Integer vector. Specify one or more page numbers to fetch. If `NULL` or `NA`, all pages will be fetched automatically.

Details

This function has two modes:

- If code = NULL, it returns a list of available biogeographical realms.
- If code is provided, it retrieves assessments for the specified realm(s), optionally filtered by year, extinction status, scope, and page(s).

If page is not specified, the function will automatically paginate over all available pages for each parameter combination.

Value

A tibble (class tbl_df``, tbl, data.frame) where each column represents a unique API response JSON key. If code = NULL, the tibble contains available biogeographical realms with columns such as realm code and name. If code is provided, the tibble contains assessment data for the specified realm(s), including taxon details, red list category, year, and other relevant metadata.

Examples

```
## Not run:
# List all available biogeographical realms
rl_biogeographical_realms()

# Retrieve all assessments for realm code 0
rl_biogeographical_realms(code = 0)

# Get latest assessments from multiple pages with filters
rl_biogeographical_realms(
  code = 0,
  year_published = c(2020, 2021),
  page = c(1, 2)
)

## End(Not run)
```

---

rl_check_api *Check IUCN Red List API Status*

Description

Verifies whether the IUCN Red List API is accessible and the provided API key is valid.

Usage

```
rl_check_api()
```

Value

Invisibly returns TRUE if the API is working properly. If not, the function will abort with an appropriate error message.

See Also

rl_set_api()

Examples

```
## Not run:
# Check if API is properly set up
rl_check_api()

## End(Not run)
```

---

rl_check_occurrences *Assessment-readiness checks for occurrence data*

Description

Run a set of data quality checks on occurrence records before they are passed to the criterion B metrics rl_eoo() and rl_aoo(). The checks flag issues that would compromise or bias the metrics. By default nothing is removed; the function reports what it finds so the assessor can decide how to proceed. Set correct to also drop the records behind the removable issues and return the cleaned data, ready to pass straight to rl_eoo() or rl_aoo().

Usage

```
rl_check_occurrences(
  x,
  coords = c("decimalLongitude", "decimalLatitude"),
  checks = NULL,
  correct = FALSE,
  recent_years = 20,
  precision_degrees = 2/111.32,
  terrestrial = TRUE,
  outlier_multiplier = 5
)
```

Arguments

x	Occurrence records: an sf POINT object (for example the output of rl_occurrences()) or a data frame with longitude and latitude columns.
coords	Character vector of length two giving the longitude and latitude column names when x is a data frame. Default c("decimalLongitude", "decimalLatitude").

checks — Character vector selecting which checks to run. Default NULL runs every check.

correct — Which removable issues to fix by dropping the offending records. FALSE (default) removes nothing and returns the report. TRUE removes every clear-error check that was run (duplicates, outliers, country, ocean_points, centroids) but not coordinate_precision, since dropping imprecise but real records is a completeness trade-off; name it explicitly to apply it. A character vector selects specific checks, including coordinate_precision. The report-only checks (unique_localities, institution_diversity, recency) describe the dataset as a whole and cannot be corrected by removing records.

recent_years — Number of years back from today within which at least one record should fall. Default 20.

precision_degrees — Coordinate precision threshold in decimal degrees. Records coarser than this (too few decimal places, or a stated uncertainty larger than this distance) are flagged. The default, 2 / 111.32 (about 0.018 degrees), corresponds to the 2 km AOO reference cell, so a record is flagged only when it genuinely cannot be placed in a 2 km grid cell.

terrestrial — Logical. Treat the taxon as terrestrial and check for records in the ocean. Default TRUE.

outlier_multiplier — Sensitivity of the outlier check: the multiplier passed to CoordinateCleaner::cc_outl(). Smaller flags more points. Default 5.

Details

The available checks are:

unique_localities fewer than 3 unique localities (EOO is undefined below 3 points). Report only.

institution_diversity all records from a single institution (possible collection bias). Report only.

recency no records within the recency window (the data may be stale). Report only.

duplicates records sharing the same coordinate, event date, and institution. Removable.

coordinate_precision coordinates coarser than a threshold, from few decimal places or a large stated uncertainty (too imprecise for the 2 by 2 km AOO grid). Removable.

outliers spatial outliers far from the main cluster, which inflate the EOO convex hull. Removable. Needs CoordinateCleaner.

country coordinates that fall outside the record's stated country (sign or transposition errors). Removable. Needs CoordinateCleaner and a countryCode column.

ocean_points records in the ocean for a terrestrial taxon. Removable. Needs CoordinateCleaner.

centroids country and capital centroids, biodiversity-institution and GBIF headquarters coordinates, and plain zeros. Removable. Needs CoordinateCleaner.

Checks needing CoordinateCleaner are skipped, with a note, when it is not installed.

Value

When `correct = FALSE`, a tibble with one row per check and the columns `check`, `status` (`"pass"`, `"warn"`, `"fail"` or `"skip"`), `n_flagged` and `detail`, returned invisibly after the results are printed. When `correct` removes issues, the cleaned occurrences are returned instead (same class as `x`), with the report attached as the `"report"` attribute.

See Also

`rl_occurrences()`, `rl_eoo()`, `rl_aoo()`

Examples

```
## Not run:
occ <- rl_occurrences("Afzelia africana", limit = 500, country = "BJ")

# Report only
rl_check_occurrences(occ)

# Clean and feed straight into a metric
clean_occ <- rl_check_occurrences(occ, correct = TRUE)
rl_aoo(clean_occ)

## End(Not run)
```

---

rl_classes *IUCN Red List taxa by class*

Description

Retrieve species assessments by taxonomic class. If `class_name = NULL`, it returns a list of available classes. If `class_name` is provided, it retrieves assessments for species in the specified class.

Usage

```
rl_classes(
  class_name = NULL,
  year_published = NULL,
  latest = NULL,
  scope_code = NULL,
  page = 1
)
```

Arguments

`class_name`	Character. The class name (e.g., "Mammalia"). Use `rl_classes()` to list available classes.
`year_published`	Optional. Single or numeric vector of years to filter assessments by publication year.

latest	Optional. Logical. If TRUE, return only the latest assessment per species.
scope_code	Optional. Integer One or more scope codes to filter assessments.
page	Optional. Integer vector. Specify one or more page numbers to fetch. If NULL or NA, all pages will be fetched automatically.

Value

A tibble (class tbl_df, tbl, data.frame) where each column represents a unique API response JSON key. If class_name = NULL, the tibble contains available taxonomic classes with a column for class names. If class_name is provided, the tibble contains assessment data for the specified class, including taxon details, red list category, year, and other relevant metadata.

Examples

```
## Not run:
# List all available classes
rl_classes()

# Get assessments for Mammalia class
rl_classes(class_name = "Mammalia")

# Get latest Aves assessments published since 2024
rl_classes(
  class_name = "Aves",
  year_published = 2024:2025,
  latest = TRUE
)

## End(Not run)
```

---

rl_comprehensive_groups

IUCN Red List Comprehensive groups

Description

Get assessment data by comprehensive group name (e.g amphibians, mammals, birds, blennies, mangrove_plants, reptiles, insects, fishes, etc). See name argument for available group names.

Usage

```
rl_comprehensive_groups(
  name = NULL,
  year_published = NULL,
  latest = NULL,
  possibly_extinct = NULL,
```

```
  possibly_extinct_in_the_wild = NULL,
  scope_code = NULL,
  page = 1
)
```

Arguments

name	Character. One or more group names. Use rl_comprehensive_groups() to list available group names.
year_published	Optional. Single or numeric vector of years to filter assessments by publication year.
latest	Optional. Logical. If TRUE, return only the latest assessment per species.
possibly_extinct	Optional. Logical. Filter for species flagged as possibly extinct.
possibly_extinct_in_the_wild	Optional. Logical. Filter for species possibly extinct in the wild.
scope_code	Optional. Integer One or more scope codes to filter assessments.
page	Optional. Integer vector. Specify one or more page numbers to fetch. If NULL or NA, all pages will be fetched automatically.

Value

A tibble (class tbl_df, tbl, data.frame) where each column represents a unique API response JSON key. If name = NULL, the tibble contains available comprehensive group names. If name is provided, the tibble contains assessment data for the specified group(s), including taxon details, red list category, year, and other relevant metadata.

Examples

```
## Not run:
rl_comprehensive_groups(name = "amphibians",
                        year_published = 2024:2025,
                        page = 1:3)

## End(Not run)
```

---

rl_conservation_actions

IUCN Red List conservation action

Description

Get assessment data by conservation action See actions argument for available action codes

Usage

```
rl_conservation_actions(
  code = NULL,
  year_published = NULL,
  latest = NULL,
  possibly_extinct = NULL,
  possibly_extinct_in_the_wild = NULL,
  scope_code = NULL,
  page = 1
)
```

Arguments

code	Character. One or more action codes Use rl_conservation_actions() to list available action codes.
year_published	Optional. Single or numeric vector of years to filter assessments by publication year.
latest	Optional. Logical. If TRUE, return only the latest assessment per species.
possibly_extinct	Optional. Logical. Filter for species flagged as possibly extinct.
possibly_extinct_in_the_wild	Optional. Logical. Filter for species possibly extinct in the wild.
scope_code	Optional. Integer One or more scope codes to filter assessments.
page	Optional. Integer vector. Specify one or more page numbers to fetch. If NULL or NA, all pages will be fetched automatically.

Value

A tibble (class tbl_df, tbl, data.frame) where each column represents a unique API response JSON key. If code = NULL, the tibble contains available conservation action codes. If code is provided, the tibble contains assessment data for the specified action code(s), including taxon details, red list category, year, and other relevant metadata.

Examples

```
## Not run:
rl_conservation_actions(code = 1,
                        year_published = 2024:2025,
                        page = 1:3)

## End(Not run)
```

`rl_countries` *Retrieve IUCN Red List assessments by country*

Description

Retrieves the species assessed by the IUCN for a specified countries. See `code` argument for available countries codes

Usage

```
rl_countries(
  code = NULL,
  year_published = NULL,
  latest = NULL,
  possibly_extinct = NULL,
  possibly_extinct_in_the_wild = NULL,
  scope_code = NULL,
  page = 1
)
```

Arguments

`code`	Character. One or more countries ISO alpha-2 code. Use `rl_countries()` to list available countries codes.
`year_published`	Optional. Single or numeric vector of years to filter assessments by publication year.
`latest`	Optional. Logical. If `TRUE`, return only the latest assessment per species.
`possibly_extinct`	Optional. Logical. Filter for species flagged as possibly extinct.
`possibly_extinct_in_the_wild`	Optional. Logical. Filter for species possibly extinct in the wild.
`scope_code`	Optional. Integer One or more scope codes to filter assessments.
`page`	Optional. Integer vector. Specify one or more page numbers to fetch. If `NULL` or `NA`, all pages will be fetched automatically.

Value

A tibble (class `tbl_df`, `tbl`, `data.frame`) where each column represents a unique API response JSON key. If `code = NULL`, the tibble contains available country codes. If `code` is provided, the tibble contains assessment data for the specified country code(s), including taxon details, red list category, year, and other relevant metadata

Examples

```
## Not run:
  # Retrieve assessments for Benin (country code "BJ") for the year 2020
  rl_countries("BJ", year = 2020)

  # Retrieve all assessments for Brazil (country code "BR")
  rl_countries("BR", page = 2)

  # Retrieve assessments for Canada (country code "CA") on specific pages
  rl_countries("CA", page = c(1, 2))

## End(Not run)
```

`rl_eoo` *Extent of occurrence (EOO) for IUCN criterion B*

Description

Compute the extent of occurrence from occurrence records as the area of the minimum convex polygon (convex hull) that encloses all points, following the IUCN Red List guidelines (section 4.9). The convex hull is the method the guidelines recommend for assessing the spatial thresholds of criterion B1.

Usage

```
rl_eoo(x, coords = c("longitude", "latitude"), crs = 4326)
```

Arguments

`x`	Occurrence records: an `sf` POINT object (for example the output of `rl_occurrences()`), or a data frame holding longitude and latitude columns.
`coords`	Character vector of length two giving the longitude and latitude column names when `x` is a data frame. Default `c("longitude", "latitude")`.
`crs`	Coordinate reference system of the input coordinates, passed to `sf`. Used only when `x` is a data frame or an `sf` object without a CRS. Default `4326` (WGS84).

Details

Geographic coordinates (longitude and latitude) are projected to a local Lambert azimuthal equal area system before the area is measured, so the result is returned in square kilometres regardless of the input coordinate system.

The extent of occurrence is undefined with fewer than three unique locations, since a polygon cannot be drawn; in that case `area_km2` is `NA` and a warning is issued. The guidelines also state that when EOO is smaller than AOO it should be raised to equal AOO; that adjustment is left to the assessor and is not applied here.

`sf` is required and is asked for interactively when it is not installed.

Value

A one row `sf` object with the convex hull in its `geometry` column (an empty polygon when fewer than three unique locations are available), returned in the input coordinate system, and the columns:

`metric` `"EOO"`.

`area_km2` the extent of occurrence in square kilometres, or `NA` when fewer than three unique locations are available.

`n_records` the number of records used.

`n_unique` the number of unique locations.

`method` the estimation method, `"convex hull"`.

`category_b1` the most threatened criterion B1 band the area reaches (`"CR"`, `"EN"` or `"VU"`), or `NA` when it reaches none (the area exceeds every band, or the metric is undefined). This is the spatial threshold only, not a full assessment, which also requires the criterion B subconditions.

References

IUCN Standards and Petitions Committee. 2024. Guidelines for Using the IUCN Red List Categories and Criteria. Version 16, section 4.9. https://cmsdocs.s3.amazonaws.com/RedListGuidelines.pdf

See Also

`rl_aoo()`

Examples

```
## Not run:
occ <- data.frame(
  longitude = c(2.1, 2.6, 3.0, 2.4, 2.9),
  latitude = c(9.1, 9.5, 9.0, 9.8, 9.3)
)
rl_eoo(occ)

## End(Not run)
```

---

rl_family *IUCN Red List taxa by family*

Description

Retrieve species assessments by taxonomic family. If `family_name = NULL`, it returns a list of available families. If `family_name` is provided, it retrieves assessments for species in the specified family.

Usage

```
rl_family(
  family_name = NULL,
  year_published = NULL,
  latest = NULL,
  scope_code = NULL,
  page = 1
)
```

Arguments

family_name	Character. The family name (e.g., "Felidae"). Use rl_family() to list available families.
year_published	Optional. Single or numeric vector of years to filter assessments by publication year.
latest	Optional. Logical. If TRUE, return only the latest assessment per species.
scope_code	Optional. Integer One or more scope codes to filter assessments.
page	Optional. Integer vector. Specify one or more page numbers to fetch. If NULL or NA, all pages will be fetched automatically.

Value

A tibble (class tbl_df, tbl, data.frame) where each column represents a unique API response JSON key. If family_name = NULL, the tibble contains available family names. If family_name is provided, the tibble contains assessment data for the specified family, including taxon details, red list category, year, and other relevant metadata.

Examples

```
## Not run:
# List all available families
rl_family()

# Get assessments for Felidae family
rl_family(family_name = "Felidae")

# Get latest Canidae assessments published from 2019 to 2022
rl_family(
  family_name = "Canidae",
  year_published = 2019:2022,
  latest = TRUE
)

## End(Not run)
```

`rl_faos` *FAO marine fishing areas*

Description

List or retrieve IUCN Red List assessments for FAO Marine Fishing Areas.

Usage

```
rl_faos(
  code = NULL,
  year_published = NULL,
  latest = NULL,
  possibly_extinct = NULL,
  possibly_extinct_in_the_wild = NULL,
  scope_code = NULL,
  page = 1
)
```

Arguments

`code`	Character. One or more FAO region codes (e.g. "21", "27"). Use `rl_faos()` with no arguments to list available FAO region codes.
`year_published`	Optional. Single or numeric vector of years to filter assessments by publication year.
`latest`	Optional. Logical. If `TRUE`, return only the latest assessment per species.
`possibly_extinct`	Optional. Logical. Filter for species flagged as possibly extinct.
`possibly_extinct_in_the_wild`	Optional. Logical. Filter for species possibly extinct in the wild.
`scope_code`	Optional. Integer One or more scope codes to filter assessments.
`page`	Optional. Integer vector. Specify one or more page numbers to fetch. If `NULL` or `NA`, all pages will be fetched automatically.

Details

If `code` is `NULL`, this returns the available FAO region codes and their descriptions. If a `code` (or multiple codes) is provided, retrieves the IUCN assessments for those regions.

Value

A tibble (class `tbl_df`, `tbl`, `data.frame`) where each column represents a unique API response JSON key. If `code = NULL`, the tibble contains available FAO region codes and their descriptions. If `code` is provided, the tibble contains assessment data for the specified FAO region(s), including description, taxon details, red list category, year, and other relevant metadata.

Examples

```
## Not run:
# List available FAO regions
rl_faos()

# Get assessments for FAO region 27
rl_faos(code = "27")

# Get assessments for regions 21 and 27 on page 1
rl_faos(code = c("21", "27"), page = 1)

## End(Not run)
```

`rl_generation_length` *Generation length for IUCN Red List assessments*

Description

Compute generation length, the average age of reproducing individuals in a population, using one of the four definitions given in the IUCN Red List guidance. Generation length sets the time window used by criteria A and C.

Usage

```
rl_generation_length(
  data,
 method = c("mean_parent", "mean_reproduction", "half_reproduction", "replacement"),
  age = NULL,
  year = NULL,
  id = NULL,
  output = NULL,
  lx = NULL,
  mx = NULL,
  na_rm = TRUE
)
```

Arguments

`data`	A data frame or tibble holding the columns the chosen `method` needs.
`method`	One of `"mean_parent"`, `"mean_reproduction"`, `"half_reproduction"`, `"replacement"`.
`age`	Column of ages, unquoted. Used by every method (as `x` in `replacement`).
`year`	Optional grouping column for `mean_parent` when the population is not at a stable age distribution. Unquoted.
`id`	Optional individual identifier for `half_reproduction`. Unquoted.
`output`	Reproductive output at each `age` for `half_reproduction`: offspring produced, or a count of breeding events as a proxy. If omitted, each row counts as one unit of output. Unquoted.

`lx`, `mx` Survivorship and age-specific fecundity columns, unquoted. Required for `method = "replacement"`, and for the life-table form of `method = "mean_reproduction"`.

`na_rm` Logical; drop missing values before computing. Default `TRUE`.

Details

Each method needs different input columns. Pass the relevant columns unquoted (tidy-style); only the columns a method actually uses need to be present.

`"mean_parent"` **(mean age of parents)** One row per newborn, holding the age of its parent when the newborn was produced. Generation length is the mean parental age. If the population is **not** at a stable age distribution, supply `year`: the mean is then taken within each year and averaged across years. Columns: `age` (required), `year` (optional).

`"mean_reproduction"` **(mean age of reproduction)** Two input shapes are accepted:

- event form: one cohort followed over life, one row per breeding event with the age at which it happened. Generation length is the mean of those ages. Columns: `age`.
- life-table form: one row per age with survivorship `lx` and age-specific fecundity `mx`. Generation length is the weighted mean age of reproduction sum(x * lx * mx) / sum(lx * mx), which reproduces the result of the official IUCN generation length calculator. Columns: `age`, `lx`, `mx`.

Supply `lx` and `mx` to use the life-table form; omit both for the event form.

`"half_reproduction"` **(50% of reproductive output)** Age at which an individual reaches half of its lifetime reproductive output, averaged over individuals. Output is measured in offspring produced; supply that as `output`. A count of breeding events works as a proxy when every event yields the same number of offspring. Two input shapes are accepted:

- event form: one row per breeding event with an `age` column (and `id` to separate individuals); each row counts as one unit of output when `output` is not supplied;
- summarised form: one row per (`id`, `age`) with an `output` column giving the offspring produced at that age.

The 50% age is the weighted median age: the smallest age at which the running total of output reaches half of the individual's lifetime total. Reproductive output happens *at* an age, so no value between age classes is invented. Columns: `age` (required), `id` (required with more than one individual), `output` (optional; omit for the one-event-per-row form).

`"replacement"` **(replacement rate)** Time for the population to grow by its net reproductive rate R0 = sum(lx * mx). With intrinsic growth rate `r` the population multiplies by exp(r * t) each unit of time, so T = log(R0) / r, where `r` solves the Euler-Lotka equation sum(exp(-r * x) * lx * mx) = 1. When the population is essentially stationary (R0 close to 1, r close to 0) the limit is the mean age of reproduction sum(x * lx * mx) / sum(lx * mx), which is returned instead. Columns: `age`, `lx`, `mx` (all required).

Value

A single numeric value: the generation length.

References

IUCN Standards and Petitions Committee. 2024. Guidelines for Using the IUCN Red List Categories and Criteria. Version 16. Prepared by the Standards and Petitions Committee. Pages 30-32. https://cmsdocs.s3.amazonaws.com/RedListGuidelines.pdf

Examples

```
# Mean age of parents at a stable age distribution
parents <- data.frame(age = c(5, 5, 3, 3, 4, 6, 6, 4, 5))
rl_generation_length(parents, "mean_parent", age = age)

# Mean age of reproduction for one cohort
cohort <- data.frame(age_of_breeding = c(5, 5, 3, 8, 5, 6, 3, 8, 9, 4))
rl_generation_length(cohort, "mean_reproduction", age = age_of_breeding)

# Same definition from a life table, reproducing the official IUCN
# calculator (this input returns 10.5, as the IUCN workbook does)
# https://www.iucnredlist.org/resources/generation-length-calculator
life_table <- data.frame(
  age = 0:19,
  lx = c(1, 0.1, rep(0.01, 17), 0),
  mx = c(0, 0, 0, rep(30, 16), 0)
)
rl_generation_length(life_table, "mean_reproduction", age = age, lx = lx, mx = mx)

# 50% of reproductive output, several mothers, weighted by breeding counts
rep_data <- data.frame(
  mother = rep(c("m1", "m2"), each = 3),
  age = c(2, 3, 4, 2, 3, 4),
  n_offspring = c(2, 3, 2, 2, 2, 2)
)
rl_generation_length(rep_data, "half_reproduction",
                     age = age, id = mother, output = n_offspring)

# Replacement rate from a life table
life_table <- data.frame(
  age = 1:4,
  lx = c(1, 0.6, 0.3, 0.1),
  mx = c(0, 1.5, 2, 1)
)
rl_generation_length(life_table, "replacement", age = age, lx = lx, mx = mx)
```

---

rl_green_status *Green status*

Description

Retrieve all IUCN Green Status assessments.

Usage

```
rl_green_status()
```

Value

A tibble (class `tbl_df`, `tbl`, `data.frame`) where each column represents a unique API response JSON key. The columns include key information about the Green Status assessment, such as year, weights, justification, and other relevant metadata.

Examples

```
## Not run:
rl_green_status()

## End(Not run)
```

---

rl_growth_forms *Growth forms*

Description

Retrieve IUCN Red List assessments by growth form.

Usage

```
rl_growth_forms(
  code = NULL,
  year_published = NULL,
  latest = NULL,
  possibly_extinct = NULL,
  possibly_extinct_in_the_wild = NULL,
  scope_code = NULL,
  page = 1
)
```

Arguments

`code`	Character. One or more growth form codes (e.g. "TREE", "SHRUB"). Use `rl_growth_forms()` with no arguments to list available growth form codes.
`year_published`	Optional. Single or numeric vector of years to filter assessments by publication year.
`latest`	Optional. Logical. If `TRUE`, return only the latest assessment per species.
`possibly_extinct`	Optional. Logical. Filter for species flagged as possibly extinct.
`possibly_extinct_in_the_wild`	Optional. Logical. Filter for species possibly extinct in the wild.

scope_code	Optional. Integer One or more scope codes to filter assessments.
page	Optional. Integer vector. Specify one or more page numbers to fetch. If NULL or NA, all pages will be fetched automatically.

Details

If code is NULL, this returns the available growth form codes and their descriptions. If a code (or multiple codes) is provided, retrieves the IUCN assessments for those growth forms.

Value

A tibble (class tbl_df, tbl, data.frame) where each column represents a unique API response JSON key. If code = NULL, the tibble contains available growth form codes and their descriptions. If code is provided, the tibble contains assessment data for the specified growth form(s), including year, taxon details, and other relevant metadata.

Examples

```
## Not run:
# List available growth form codes
rl_growth_forms()

# Get assessments for tree growth form (e.g Geophyte)
rl_growth_forms(code = "GE")

# Get assessments for multiple growth forms (e.g Hydrophyte, Lithophyte)
rl_growth_forms(code = c("H", "L"), page = c(1, 2))

## End(Not run)
```

---

rl_habitats *Habitats*

Description

Retrieve IUCN Red List assessments by habitat classification.

Usage

```
rl_habitats(
  code = NULL,
  year_published = NULL,
  latest = NULL,
  possibly_extinct = NULL,
  possibly_extinct_in_the_wild = NULL,
  scope_code = NULL,
  page = 1
)
```

Arguments

code	Character. One or more habitat classification codes. Use rl_habitats() with no arguments to list available habitat codes.
year_published	Optional. Single or numeric vector of years to filter assessments by publication year.
latest	Optional. Logical. If TRUE, return only the latest assessment per species.
possibly_extinct	Optional. Logical. Filter for species flagged as possibly extinct.
possibly_extinct_in_the_wild	Optional. Logical. Filter for species possibly extinct in the wild.
scope_code	Optional. Integer One or more scope codes to filter assessments.
page	Optional. Integer vector. Specify one or more page numbers to fetch. If NULL or NA, all pages will be fetched automatically.

Value

A tibble (class tbl_df, tbl, data.frame) where each column represents a unique API response JSON key. If code = NULL, the tibble contains available habitat codes and their descriptions. If code is provided, the tibble contains assessment data for the specified habitat(s), including taxon details, description, red list category, year, assessment id, and other relevant metadata.

Examples

```
## Not run:
# Retrieve available habitat codes
rl_habitats()

# Retrieve assessments for the Desert
rl_habitats(code = 8)

## End(Not run)
```

---

rl_kingdoms *IUCN Red List taxa by kingdom*

Description

Retrieve species assessments by kingdom. If kingdom_name = NULL, it returns a list of available kingdoms. If kingdom_name is provided, it retrieves assessments for species in the specified kingdom.

Usage

```
rl_kingdoms(
  kingdom_name = NULL,
  year_published = NULL,
  latest = NULL,
  scope_code = NULL,
  page = 1
)
```

Arguments

kingdom_name	Character. The kingdom name (e.g., "Animalia"). Use rl_kingdoms() to list available kingdoms.
year_published	Optional. Single or numeric vector of years to filter assessments by publication year.
latest	Optional. Logical. If TRUE, return only the latest assessment per species.
scope_code	Optional. Integer One or more scope codes to filter assessments.
page	Optional. Integer vector. Specify one or more page numbers to fetch. If NULL or NA, all pages will be fetched automatically.

Value

A tibble (class tbl_df, tbl, data.frame) where each column represents a unique API response JSON key. If kingdom_name = NULL, the tibble contains available kingdom names. If kingdom_name is provided, the tibble contains assessment data for the specified kingdom, including taxon details, red list category, year, and other relevant metadata.

Examples

```
## Not run:
# List all available kingdoms
rl_kingdoms()

# Get assessments for species in Animalia kingdom
rl_kingdoms(kingdom_name = "Animalia")

# Get latest assessments for Plantae published in 2021
rl_kingdoms(
  kingdom_name = "Plantae",
  year_published = 2021,
  latest = TRUE
)

## End(Not run)
```

`rl_name_resolve` *Resolve a scientific name to the accepted IUCN Red List name*

Description

When you work with names from another source (for example GBIF), some names differ from the ones the IUCN Red List uses, so a direct request such as `rl_scientific_name()` returns *404 (species not found)*. This function forces a match: it verifies the name against the GlobalNames Verifier API and returns the accepted (current) name the IUCN Red List uses, so the retrieval can be repeated with the resolved name.

Usage

```
rl_name_resolve(
  genus_name,
  species_name,
  infra_name = NULL,
  subpopulation_name = NULL,
  vernaculars = "all",
  data_sources = 163,
  all_matches = FALSE,
  capitalize = FALSE,
  species_group = FALSE,
  fuzzy_uninomial = FALSE,
  stats = FALSE,
  main_taxon_threshold = 0.6
)
```

Arguments

`genus_name`	Character. The genus name (required).
`species_name`	Character. The species name (required).
`infra_name`	Character. The infraspecific name (optional).
`subpopulation_name`	Character. The subpopulation name (optional).
`vernaculars`	Character. Vernacular-name languages to return, as `"|"`-separated ISO 639-3 codes (e.g. "eng|rus|deu" for multiple languages), or `"all"` for every language found. Default `"all"`.
`data_sources`	Integer vector of GlobalNames data-source ids to match against. Default `163` for IUCN. See `https://verifier.globalnames.org/data_sources` for the full list.
`all_matches`	Logical. If `TRUE`, return every match found instead of only the best one. Default `FALSE`.
`capitalize`	Logical. Capitalize the first letter of the name before matching. Default `FALSE`.

species_group Logical. Expand the search to the species group where applicable. Default FALSE.

fuzzy_uninomial
Logical. Allow fuzzy matching for uninomial names. Default FALSE.

stats Logical. Ask the API to find the kingdom and main taxon holding most names (Catalogue of Life only). Default FALSE.

main_taxon_threshold
Numeric between 0.5 and 1 setting the minimal proportion for main-taxon discovery. Default 0.6.

Value

A tibble with one row per input name. A name that cannot be matched returns a single row of NA fields.

See Also

rl_scientific_name()

Examples

```
## Not run:
# GBIF's "Corvinella corvina" is a synonym; IUCN uses "Lanius corvinus"
rl_name_resolve(genus_name = "Corvinella", species_name = "corvina")

## End(Not run)
```

---

rl_occurrences *Retrieve GBIF occurrence records for a taxon*

Description

Fetch occurrence records from the Global Biodiversity Information Facility (GBIF) and return them as a clean sf POINT object ready for the criterion B metrics rl_eoo() and rl_aoo(). This uses the public GBIF search API through rgbif, so no GBIF account, username, or password is needed. Only bulk downloads (the GBIF download API) require credentials, and those are not used here.

Usage

```
rl_occurrences(
  x,
  limit = 500,
  country = NULL,
  year = NULL,
  basis_of_record = NULL,
  has_coordinate = TRUE,
  has_geospatial_issue = FALSE,
```

```
  correct = NULL,
  progress = TRUE,
  crs = 4326,
  ...
)
```

Arguments

x	The taxon to retrieve. One of: • a scientific name, for example "Afzelia africana"; • a GBIF backbone taxon key (a number); • a data frame from a name resolution step (for example the output of rl_name_resolve() or rl_scientific_name()), from which a name column is detected.
limit	Maximum number of records to return. Default 500. Use Inf to fetch every available record. The search API is paged in blocks of 300 behind the scenes, up to its ceiling of 100000 records; beyond that a GBIF download (which needs an account) would be required.
country	Optional ISO 3166-1 alpha-2 country code to restrict records, for example "BJ" for Benin. Pass a vector for several countries (matched as OR), for example c("BJ", "NG").
year	Optional year filter. A single year (2000), a "min,max" range string ("2000,2020", open-ended as "2000,*" or "*,2000"), or a comparator string (">2025", ">=2025", "<2000", "<=2000"). A range is one comma-separated string, not a vector.
basis_of_record	Optional GBIF basis of record filter, for example "HUMAN_OBSERVATION" or "PRESERVED_SPECIMEN". Pass a vector for several types (matched as OR), for example c("HUMAN_OBSERVATION", "MACHINE_OBSERVATION").
has_coordinate	Logical. Keep only records that carry coordinates. Default TRUE.
has_geospatial_issue	Logical. Keep records that GBIF flags with a geospatial issue. Default FALSE (drop flagged records).
correct	Which readiness checks from rl_check_occurrences() to run after the download finishes, and apply. One of: • NULL (default): run no checks. • TRUE: run every check and drop the records behind the clear-error issues (duplicates, outliers, country, ocean_points, centroids). • a character vector of check names (for example c("outliers", "duplicates")): run those and drop the records they flag. coordinate_precision is always reported but never applied here, since dropping imprecise but real records can gut the sample and bias the metrics; to apply it deliberately, call rl_check_occurrences() with correct = "coordinate_precision". The report-only checks (unique_localities, institution_diversity, recency) are reported but not corrected. When any correction removes records the cleaned sf is returned; otherwise the readiness report is attached to the returned sf as its "report" attribute. For finer

	control (thresholds, report without correcting), call `rl_check_occurrences()` directly.
`progress`	Logical. Show a progress bar while records are downloaded. Default `TRUE`.
`crs`	Coordinate reference system for the returned `sf` object. Default `4326` (WGS84), the system GBIF coordinates use.
`...`	Further named filters passed straight to `rgbif::occ_search()`, for example `continent`, `institutionCode`, `elevation`, or `coordinateUncertaintyInMeters`.

Details

Because records are queried by the GBIF backbone taxon key, occurrences that GBIF indexes under synonyms of the accepted name are already included.

Value

An `sf` POINT object (WGS84 by default) with one row per occurrence record and the GBIF fields returned by the search, such as `scientificName`, `eventDate`, `year`, `country`, `basisOfRecord`, `institutionCode`, and `coordinateUncertaintyInMeters`. When no record matches, an empty `sf` object is returned with a warning. Records with invalid coordinates (missing, out of range, null island, or absence records) are always dropped; further data quality checks run only when `correct` is set.

See Also

`rl_check_occurrences()`, `rl_eoo()`, `rl_aoo()`, `rl_name_resolve()`

Examples

```
## Not run:
# By scientific name, capped at 300 records, Benin only
occ <- rl_occurrences("Afzelia africana", limit = 300, country = "BJ")

# Pass any GBIF filter through `...`
occ <- rl_occurrences("Panthera leo", year = "2010,2020",
                      basis_of_record = "HUMAN_OBSERVATION")

# Download, then run and correct specific checks
occ <- rl_occurrences("Afzelia africana", limit = 500, country = "BJ",
                      correct = c("outliers", "duplicates"))

# Straight into a criterion B metric
rl_eoo(occ)

## End(Not run)
```

rl_open_file *Open file for editing*

Description

Opens a specified file for editing in the system's default editor (as configured by R).

Usage

```
rl_open_file(path = NULL, scope = c("user", "project"))
```

Arguments

path	Optional character string specifying the path to the file to open. If NULL (default), a .Renviron file is opened based on the value of scope.
scope	Character string indicating which .Renviron file to open when path = NULL: • user: Opens the user-level .Renviron • project: Opens or creates a .Renviron file in the current working directory

Value

(Invisibly) returns the path to the file opened.

Examples

```
## Not run:
# Open user-level .Renviron
open_file()

## End(Not run)
```

rl_orders *IUCN Red List taxa by order*

Description

Retrieve species assessments by taxonomic order. If order_name = NULL, it returns a list of available orders. If order_name is provided, it retrieves assessments for species in the specified order.

Usage

```
rl_orders(
  order_name = NULL,
  year_published = NULL,
  latest = NULL,
  scope_code = NULL,
  page = 1
)
```

Arguments

order_name	Character. The order name (e.g., "Carnivora"). Use rl_orders() to list available orders.
year_published	Optional. Single or numeric vector of years to filter assessments by publication year.
latest	Optional. Logical. If TRUE, return only the latest assessment per species.
scope_code	Optional. Integer One or more scope codes to filter assessments.
page	Optional. Integer vector. Specify one or more page numbers to fetch. If NULL or NA, all pages will be fetched automatically.

Value

A tibble (class tbl_df, tbl, data.frame) where each column represents a unique API response JSON key. If order_name = NULL, the tibble contains available taxonomic orders with a column for order names. If order_name is provided, the tibble contains assessment data for the specified order, including year, taxon details, criteria, and other relevant metadata.

Examples

```
## Not run:
# List all available orders
rl_orders()

# Get assessments for Carnivora order
rl_orders(order_name = "Carnivora")

# Get latest Primates assessments published in 2022
rl_orders(
  order_name = "Primates",
  year_published = 2022,
  latest = TRUE
)

## End(Not run)
```

`rl_overall_reduction` *Overall reduction across subpopulations for IUCN criterion A*

Description

Combine the reductions of several subpopulations into one reduction for the taxon, following section 4.5.4 of the IUCN Red List guidelines. The overall reduction is the change in the summed population, which equals the average of the subpopulation reductions weighted by their size three generations ago.

Usage

```
rl_overall_reduction(
  past = NULL,
  present = NULL,
  reduction = NULL,
  subpopulation = NULL,
  subcriterion = c("A2", "A1", "A3", "A4")
)
```

Arguments

`past`	Numeric vector of subpopulation sizes at the start of the window (three generations ago).
`present`	Numeric vector of subpopulation sizes at the end of the window.
`reduction`	Numeric vector of subpopulation reductions (proportions).
`subpopulation`	Optional labels for the subpopulations.
`subcriterion`	Which criterion A subcriterion sets the thresholds for `category_a`: `"A2"` (default), `"A1"`, `"A3"` or `"A4"`.

Details

Give any two of `past`, `present` and `reduction` for each subpopulation and the third is worked out from `reduction = 1 - present / past`. The sizes should already be projected to the start and end of the same window, for example with `rl_reduction()`.

Value

A one row tibble with the overall `reduction` and `reduction_pct`, the number of subpopulations, the summed past and present sizes, and `category_a`. The per-subpopulation table is attached as the `"subpopulations"` attribute.

References

IUCN Standards and Petitions Committee. 2024. Guidelines for Using the IUCN Red List Categories and Criteria. Version 16, section 4.5.4.

See Also

rl_reduction()

Examples

```
# Guidelines Example 1: past and present sizes for three subpopulations
rl_overall_reduction(
  past = c(10000, 8000, 12000),
  present = c(5000, 9000, 2000),
  subpopulation = c("Pacific", "Atlantic", "Indian")
)
```

---

rl_phylum *IUCN Red List taxa by phylum*

Description

Retrieve species assessments by phylum. If phylum_name = NULL, it returns a list of available phyla. If phylum_name is provided, it retrieves assessments for species in the specified phylum.

Usage

```
rl_phylum(
  phylum_name = NULL,
  year_published = NULL,
  latest = NULL,
  scope_code = NULL,
  page = 1
)
```

Arguments

phylum_name	Character. The phylum name (e.g., "Chordata"). Use rl_phylum() to list available phyla.
year_published	Optional. Single or numeric vector of years to filter assessments by publication year.
latest	Optional. Logical. If TRUE, return only the latest assessment per species.
scope_code	Optional. Integer One or more scope codes to filter assessments.
page	Optional. Integer vector. Specify one or more page numbers to fetch. If NULL or NA, all pages will be fetched automatically.

Value

A tibble (class tbl_df, tbl, data.frame) where each column represents a unique API response JSON key. If phylum_name = NULL, the tibble contains available phylum names. If phylum_name is provided, the tibble contains assessment data for the specified phylum, year, latest, possibly extincts, and other relevant metadata.

Examples

```
## Not run:
# List all available phyla
rl_phylum()

# Get assessments for species in Chordata phylum
rl_phylum(phylum_name = "Chordata")

# Get latest assessments for Arthropoda published in 2020
rl_phylum(
  phylum_name = "Arthropoda",
  year_published = 2020,
  latest = TRUE
)

## End(Not run)
```

`rl_population_trends` *IUCN Red List population trends*

Description

Retrieve available population trend categories or species assessments for one or more trends.

Usage

```
rl_population_trends(
  code = NULL,
  year_published = NULL,
  latest = NULL,
  possibly_extinct = NULL,
  possibly_extinct_in_the_wild = NULL,
  scope_code = NULL,
  page = 1
)
```

Arguments

`code`	Character or Numeric. One or more population trend codes (`0`-`3`). Use `rl_population_trends()` to list available trend codes and definition.
`year_published`	Optional. Single or numeric vector of years to filter assessments by publication year.
`latest`	Optional. Logical. If `TRUE`, return only the latest assessment per species.
`possibly_extinct`	Optional. Logical. Filter for species flagged as possibly extinct.
`possibly_extinct_in_the_wild`	Optional. Logical. Filter for species possibly extinct in the wild.

scope_code	Optional. Integer One or more scope codes to filter assessments.
page	Optional. Integer vector. Specify one or more page numbers to fetch. If NULL or NA, all pages will be fetched automatically.

Details

This function has two modes:

- If code = NULL, it returns a list of available population trend categories.
- If code is provided, it retrieves assessments for the specified trend(s), optionally filtered by year, extinction status, scope, and page(s).

Population trends include: Increasing, Decreasing, Stable, or Unknown.

If page is not specified, the function will automatically paginate over all available pages for each parameter combination.

Value

A tibble (class tbl_df, tbl, data.frame) where each column represents a unique API response JSON key. If code = NULL, the tibble contains available population trend categories with columns such as code and description. If code is provided, the tibble contains assessment data for the specified trend(s), including population trend description, population trend code, year, latest, and other relevant metadata.

Examples

```
## Not run:
# List all available population trend categories
rl_population_trends()

# Retrieve assessments for species with decreasing populations
rl_population_trends(code = "1")

# Get latest decreasing population assessments from 2020
rl_population_trends(
  code = 2,
  year_published = 2020,
  latest = TRUE
)

## End(Not run)
```

---

rl_possibly_extinct *IUCN Red List possibly extinct taxa*

Description

Retrieve species assessments flagged as possibly extinct. Returns all latest global assessments for taxa that are possibly extinct.

Usage

```
rl_possibly_extinct()
```

Value

A tibble (class `tbl_df`, `tbl`, `data.frame`) where each column represents a unique API response JSON key. Columns include year, latest, possibly extinct, possibly extinct in the wild, sis taxon id, url, taxon scientific name, red list category, assessment id, scopes description, and scopes code

Examples

```
## Not run:
# Get all possibly extinct species
rl_possibly_extinct()

## End(Not run)
```

---

rl_possibly_extinct_in_wild

IUCN Red List possibly extinct in the wild taxa

Description

Retrieve species assessments flagged as possibly extinct in the wild. Returns all latest global assessments for taxa that are possibly extinct in the wild.

Usage

```
rl_possibly_extinct_in_wild()
```

Value

A tibble (class `tbl_df`, `tbl`, `data.frame`) where each column represents a unique API response JSON key. Columns include year, latest, possibly extinct, possibly extinct in the wild, sis taxon id, url, taxon scientific name, red list category, assessment id, scopes description, and scopes code

Examples

```
## Not run:
# Get all possibly extinct in the wild species
rl_possibly_extinct_in_wild()

## End(Not run)
```

`rl_red_list_categories` *IUCN Red List Categories*

Description

Retrieve species assessments based on their Red List threat categories. If `code = NULL`, it returns a list of available Red List categories. If `code` is provided, it retrieves assessments for species in the specified category(ies).

Usage

```
rl_red_list_categories(
  code = NULL,
  year_published = NULL,
  latest = NULL,
  possibly_extinct = NULL,
  possibly_extinct_in_the_wild = NULL,
  scope_code = NULL,
  page = 1
)
```

Arguments

`code`	Character. One or more Red List category codes (e.g., "CR", "EN"). Use `rl_red_list_categories()` to list available categories.
`year_published`	Optional. Single or numeric vector of years to filter assessments by publication year.
`latest`	Optional. Logical. If `TRUE`, return only the latest assessment per species.
`possibly_extinct`	Optional. Logical. Filter for species flagged as possibly extinct.
`possibly_extinct_in_the_wild`	Optional. Logical. Filter for species possibly extinct in the wild.
`scope_code`	Optional. Integer One or more scope codes to filter assessments.
`page`	Optional. Integer vector. Specify one or more page numbers to fetch. If `NULL` or `NA`, all pages will be fetched automatically.

Value

A tibble (class `tbl_df`, `tbl`, `data.frame`) where each column represents a unique API response JSON key. If `code = NULL`, the tibble contains available Red List categories with columns such as code and description. If `code` is provided, the tibble contains assessment data for the specified category(ies), including year, taxon details, and other relevant metadata.

Examples

```
## Not run:
# List all Red List categories
rl_red_list_categories()

# Get Critically Endangered species assessments
rl_red_list_categories(code = "CR")

# Get Vulnerable species assessments published in 2020
rl_red_list_categories(
  code = "VU",
  year_published = 2020
)

## End(Not run)
```

`rl_reduction` *Population reduction for IUCN criterion A*

Description

Estimate the population reduction of a taxon over the most recent three generations (or ten years, whichever is longer), following section 4.5 of the IUCN Red List guidelines. A decline model is fitted to the population estimates and used to read off the population size at the start and end of the window, and the reduction is the proportional drop between them. The same calculation gives the estimated continuing decline of criterion C1 and B when a different window is set through `years`.

Usage

```
rl_reduction(
  population,
  time,
  generation_length,
  model = c("exponential", "linear"),
  assessment_year = NULL,
  years = NULL,
  subcriterion = c("A2", "A1", "A3", "A4")
)
```

Arguments

`population`	Numeric vector of population sizes (number of mature individuals, or an index that scales with it).
`time`	Numeric vector of the years the sizes refer to, the same length as `population`.
`generation_length`	Generation length in years. See `rl_generation_length()`.
`model`	Decline pattern, `"exponential"` (default) or `"linear"`.

`assessment_year`
: The year taken as the present. Default is the most recent year in `time`.

`years`
: Length of the assessment window in years. Default is the longer of three generations or ten years.

`subcriterion`
: Which criterion A subcriterion sets the thresholds for `category_a`: `"A2"` (default), `"A1"`, `"A3"` or `"A4"`.

Details

Two decline patterns are supported.

`"exponential"` a constant proportional rate of decline, fitted as a log-linear regression of population size on time. Appropriate when the rate of loss stays proportional to population size, for example a constant harvest fraction.

`"linear"` a constant number of individuals lost per year, fitted as a linear regression of population size on time. Appropriate when a fixed amount is removed each year, for example a fixed area of habitat lost.

With exactly two estimates the fit passes through both points, reproducing the two-point formulas in the guidelines. With more estimates the regression smooths natural variation, and the reduction is still read over the most recent window.

The `category_a` column reports the most threatened band the reduction reaches for the chosen subcriterion, or `NA` when it reaches none (including an increase). The criterion A thresholds are 50/70/90 percent (VU/EN/CR) for `"A1"` and 30/50/80 percent for `"A2"`, `"A3"` and `"A4"`. This is the magnitude threshold only, not a full assessment.

Value

A one row tibble with the model and window used, the fitted population sizes at the start and end of the window (`n_start`, `n_present`), the `reduction` (a proportion) and `reduction_pct`, and `category_a`.

References

IUCN Standards and Petitions Committee. 2024. Guidelines for Using the IUCN Red List Categories and Criteria. Version 16, section 4.5. https://cmsdocs.s3.amazonaws.com/RedListGuidelines.pdf

See Also

`rl_overall_reduction()`, `rl_generation_length()`

Examples

```
# Guidelines example: 20000 in 1961 and 14000 in 1981, generation length 20,
# assessed in 2001 so the three generation window runs 1941 to 2001.
# Exponential decline gives a 65.7 percent reduction.
rl_reduction(population = c(20000, 14000), time = c(1961, 1981),
             generation_length = 20, model = "exponential",
             assessment_year = 2001)
```

```
# The same data under a linear decline gives 69.2 percent.
rl_reduction(population = c(20000, 14000), time = c(1961, 1981),
             generation_length = 20, model = "linear",
             assessment_year = 2001)
```

`rl_research` *IUCN Red List research categories*

Description

Retrieve species assessments based on their research needs categories. If `code = NULL`, it returns a list of available research categories. If `code` is provided, it retrieves assessments for species with the specified research need(s).

Usage

```
rl_research(
  code = NULL,
  year_published = NULL,
  latest = NULL,
  possibly_extinct = NULL,
  possibly_extinct_in_the_wild = NULL,
  scope_code = NULL,
  page = 1
)
```

Arguments

`code`	Character Or Numeric. One or more research category codes (e.g., "1", "2"). Use `rl_research()` to list available research categories.
`year_published`	Optional. Single or numeric vector of years to filter assessments by publication year.
`latest`	Optional. Logical. If `TRUE`, return only the latest assessment per species.
`possibly_extinct`	Optional. Logical. Filter for species flagged as possibly extinct.
`possibly_extinct_in_the_wild`	Optional. Logical. Filter for species possibly extinct in the wild.
`scope_code`	Optional. Integer One or more scope codes to filter assessments.
`page`	Optional. Integer vector. Specify one or more page numbers to fetch. If `NULL` or `NA`, all pages will be fetched automatically.

Value

A tibble (class `tbl_df`, `tbl`, `data.frame`) where each column represents a unique API response JSON key. If `code = NULL`, the tibble contains available research categories with columns such as code and description. If `code` is provided, the tibble contains assessment data for the specified research need(s), including description, research code, year, taxon details, and other relevant metadata.

Examples

```
## Not run:
# List all research categories
rl_research()

# Get species needing population trends research (code 3_1)
rl_research(code = "3_1")

# Get species needing life history & ecology research published since 2019
rl_research(
  code = "1_3",
  year_published = 2019:2023
)

## End(Not run)
```

`rl_scientific_name` *IUCN Red List taxa by scientific name*

Description

Retrieve species assessments using scientific names (Latin binomials). Returns summary assessment data including both latest and historic assessments.

Usage

```
rl_scientific_name(
  genus_name,
  species_name,
  infra_name = NULL,
  subpopulation_name = NULL,
  resolve = TRUE
)
```

Arguments

`genus_name`	Character. The genus name (required).
`species_name`	Character. The species name (required).
`infra_name`	Character. The infraspecific name (optional).

subpopulation_name
: Character. The subpopulation name (optional).

resolve
: Logical. If TRUE (default), attempt to resolve the name via rl_name_resolve() and retry when the IUCN Red List returns a 404 (species not found). Set to FALSE to fail instead.

Details

When the supplied name is not the one the IUCN Red List uses (a common case for names coming from GBIF), the API returns a *404* and, if resolve = TRUE, this function calls rl_name_resolve() to recover the accepted IUCN name and retries the request with it. The resolved-name provenance is then appended as the extra columns outlink, entryDate, currentCanonicalFull, isSynonym and matchType.

Value

A tibble (class tbl_df, tbl, data.frame) where each column represents a unique API response JSON key. The tibble contains assessment data for the specified taxon, including taxon details. When the name had to be resolved, the columns input_name, isSynonym, entryDate, and matchType from rl_name_resolve() are appended.

See Also

rl_name_resolve()

Examples

```
## Not run:
# Get assessments for Panthera leo (lion)
rl_scientific_name(genus_name = "Panthera", species_name = "leo")

# A GBIF synonym that IUCN lists under another name is resolved automatically
rl_scientific_name(genus_name = "Corvinella", species_name = "corvina")

## End(Not run)
```

---

rl_scopes *IUCN Red List assessment scopes*

Description

Retrieve species assessments based on their geographic assessment scopes. If code = NULL, it returns a list of available assessment scopes. If code is provided, it retrieves assessments for the specified scope(s).

Usage

```
rl_scopes(
  code = NULL,
  year_published = NULL,
  latest = NULL,
  possibly_extinct = NULL,
  possibly_extinct_in_the_wild = NULL,
  scope_code = scope_code %||% NA,
  page = 1
)
```

Arguments

code	Character or Numeric. One or more scope codes (e.g., "1", "2"). Use rl_scopes() to list available scope categories.
year_published	Optional. Single or numeric vector of years to filter assessments by publication year.
latest	Optional. Logical. If TRUE, return only the latest assessment per species.
possibly_extinct	Optional. Logical. Filter for species flagged as possibly extinct.
possibly_extinct_in_the_wild	Optional. Logical. Filter for species possibly extinct in the wild.
scope_code	Optional. Integer One or more scope codes to filter assessments.
page	Optional. Integer vector. Specify one or more page numbers to fetch. If NULL or NA, all pages will be fetched automatically.

Value

A tibble (class tbl_df, tbl, data.frame) where each column represents a unique API response JSON key. If code = NULL, the tibble contains available assessment scopes with columns such as code and description. If code is provided, the tibble contains assessment data for the specified scope(s), including, description, year, latest, taxon details, and other relevant metadata.

Examples

```
## Not run:
# List all assessment scopes
rl_scopes()

# Get globally assessed species (code 1)
rl_scopes(code = "1")

# Get Pan-Africa species assessed species published since 2020
rl_scopes(
  code = "2",
  year_published = 2020:2023
)

## End(Not run)
```

rl_set_api *Set the IUCN Red List API key*

Description

The function provide steps to set the IUCN Red List API key.

Usage

```
rl_set_api(api_key)
```

Arguments

api_key	Character. The API key provided by the IUCN Red List to authenticate requests, obtainable at IUCN Red List API website.

Value

Invisibly returns NULL after setting the API key.

Examples

```
## Not run:
# Set the API key for the IUCN Red List
rl_set_api("your_api_key")

## End(Not run)
```

rl_sis *IUCN Red List taxa by SIS ID*

Description

Retrieve species assessments using the Species Information Service (SIS) identifier. Returns summary assessment data including both latest and historic assessments.

Usage

```
rl_sis(sis_id = 179359)
```

Arguments

sis_id	Numeric. One or more SIS identifiers for taxa.

Value

A tibble (class `tbl_df`, `tbl`, `data.frame`) containing assessment data for the specified SIS ID(s).

Examples

```
## Not run:
# Get assessments for species with SIS ID 179359
rl_sis(179359)

## End(Not run)
```

`rl_statistics` *IUCN Red List statistics*

Description

Retrieve count of species with assessments. This endpoint returns the total number of assessed species on the IUCN Red List.

Usage

```
rl_statistics()
```

Value

A tibble (class `tbl_df`, `tbl`, `data.frame`) containing count and date of access.

Examples

```
## Not run:
# Get total count of assessed species
rl_statistics()

## End(Not run)
```

`rl_stresses` *IUCN Red List stress categories*

Description

Retrieve species assessments based on stress categories affecting species. If `code = NULL`, it returns a list of available stress categories. If `code` is provided, it retrieves assessments for species affected by the specified stress(es).

Usage

```
rl_stresses(
  code = NULL,
  year_published = NULL,
  latest = NULL,
  possibly_extinct = NULL,
  possibly_extinct_in_the_wild = NULL,
  scope_code = NULL,
  page = 1
)
```

Arguments

code	Character or Numeric. One or more stress codes (e.g., "1", "2_1"). Use rl_stresses() to list available stress categories.
year_published	Optional. Single or numeric vector of years to filter assessments by publication year.
latest	Optional. Logical. If TRUE, return only the latest assessment per species.
possibly_extinct	Optional. Logical. Filter for species flagged as possibly extinct.
possibly_extinct_in_the_wild	Optional. Logical. Filter for species possibly extinct in the wild.
scope_code	Optional. Integer One or more scope codes to filter assessments.
page	Optional. Integer vector. Specify one or more page numbers to fetch. If NULL or NA, all pages will be fetched automatically.

Value

A tibble (class tbl_df, tbl, data.frame) containing stress categories or species assessments. If code = NULL, the tibble contains available stress categories with columns such as code and description. If code is provided, the tibble contains assessment data for the specified stress(es), including year, taxon details, and other relevant metadata.

Examples

```
## Not run:
# List all stress categories
rl_stresses()

# Get species affected by ecosystem stresses (code 1)
rl_stresses(code = "1") # or code = 1

# Get species affected by competition stresses published since 2020
rl_stresses(
  code = "2_3_2",
  year_published = 2020:2023
)

## End(Not run)
```

`rl_systems` *IUCN Red List ecological systems*

Description

Retrieve species assessments based on their ecological systems. If `code = NULL`, it returns a list of available ecological systems. If `code` is provided, it retrieves assessments for species in the specified system(s).

Usage

```
rl_systems(
  code = NULL,
  year_published = NULL,
  latest = NULL,
  possibly_extinct = NULL,
  possibly_extinct_in_the_wild = NULL,
  scope_code = NULL,
  page = 1
)
```

Arguments

`code`	Character or Numeric. One or more system codes (e.g., "0", "1", "2"). Use `rl_systems()` to list available ecological systems.
`year_published`	Optional. Single or numeric vector of years to filter assessments by publication year.
`latest`	Optional. Logical. If `TRUE`, return only the latest assessment per species.
`possibly_extinct`	Optional. Logical. Filter for species flagged as possibly extinct.
`possibly_extinct_in_the_wild`	Optional. Logical. Filter for species possibly extinct in the wild.
`scope_code`	Optional. Integer One or more scope codes to filter assessments.
`page`	Optional. Integer vector. Specify one or more page numbers to fetch. If `NULL` or `NA`, all pages will be fetched automatically.

Value

A tibble (class `tbl_df`, `tbl`, `data.frame`) where each column represents a unique API response JSON key. If `code = NULL`, the tibble contains available ecological systems with columns such as code and description. If `code` is provided, the tibble contains assessment data for the specified system(s), including description, possible extinct in the wild, scientific name, latest, taxon details, and other relevant metadata.

Examples

```
## Not run:
# List all ecological systems
rl_systems()

# Get terrestrial species assessments (code 0)
rl_systems(code = 0)

# Get marine species assessments published since 2021
rl_systems(
  code = "2",
  year_published = 2021:2023
)

## End(Not run)
```

`rl_threat_category` *Find the IUCN Red List threat category interactively*

Description

An interactive decision tree that walks you through the five IUCN Red List criteria (A to E) like a form filled in at the console, and tells you which threatened category (Critically Endangered, Endangered or Vulnerable) your values meet. You can go back to any earlier question to change an answer, and you review all your answers before the result is shown.

Usage

```
rl_threat_category()
```

Details

Start by choosing the criteria you have data for; only the questions for those criteria are asked. At every prompt you can type:

`b` go back to the previous question;

`q` quit without a result;

`?` show help for the current question;

Enter keep your current answer when you come back to a question, or leave an optional value unknown;

`na` set an optional value back to unknown.

The thresholds are those of the summary of the five criteria in the IUCN Red List Categories and Criteria:

Criterion	CR	EN	VU
A1 reduction	>= 90%	>= 70%	>= 50%

A2, A3, A4 reduction	>= 80%	>= 50%	>= 30%
B1 extent of occurrence	< 100 km2	< 5,000 km2	< 20,000 k
B2 area of occupancy	< 10 km2	< 500 km2	< 2,000 k
B(a) number of locations	1	<= 5	<= 10
C mature individuals	< 250	< 2,500	< 10,000
C2a(i) largest subpopulation	<= 50	<= 250	<= 1,000
C2a(ii) share in one subpopulation	90-100%	95-100%	100%
D mature individuals	< 50	< 250	< 1,000 (D
E probability of extinction	>= 50% in 10 years or 3 generations	>= 20% in 20 years or 5 generations	>= 10% in

Criterion B also needs at least two of the conditions (a) severe fragmentation or a limited number of locations, (b) continuing decline and (c) extreme fluctuations. Criterion C also needs a continuing decline (C1) or a continuing decline with a small or concentrated subpopulation structure or extreme fluctuations (C2). D2 (a very restricted range facing a plausible, fast-acting threat) applies to Vulnerable only. When both are given and the extent of occurrence is smaller than the area of occupancy, the extent of occurrence is raised to equal it, as the guidelines require.

A taxon is listed in the most threatened category that any criterion qualifies it for. When no criterion is met, the taxon does not qualify for a threatened category on the information given; it may be Near Threatened, Least Concern or Data Deficient, which this function does not decide.

This is a screening aid, not a formal assessment. Read the definitions of the terms used (location, severe fragmentation, continuing decline, extreme fluctuations) in the IUCN guidelines before listing a taxon.

Value

Invisibly, an object of class `rl_threat_category`: a list with

`category` `"CR"`, `"EN"`, `"VU"`, or `NA` when no threatened category is met;

`criteria` the criteria met at that category, for example `"A2; B1ab"`, or `NA`;

`details` a tibble with one row per criterion assessed and the columns `criterion`, `category`, `code` and `reason`;

`answers` a list of the values you entered.

Printing the object shows the report again. If you quit, `NULL` is returned invisibly.

References

IUCN. 2012. IUCN Red List Categories and Criteria: Version 3.1. Second edition. Gland, Switzerland and Cambridge, UK: IUCN.

IUCN Standards and Petitions Committee. 2024. Guidelines for Using the IUCN Red List Categories and Criteria. Version 16. https://cmsdocs.s3.amazonaws.com/RedListGuidelines.pdf

See Also

`rl_reduction()` to estimate a population reduction for criterion A, `rl_eoo()` and `rl_aoo()` for the criterion B range metrics, and `rl_generation_length()`.

Examples

```
## Not run:
res <- rl_threat_category()
res$category
res$details

## End(Not run)
```

`rl_threats` *IUCN Red List threat categories*

Description

Retrieve species assessments based on threat categories. If `code = NULL`, it returns a list of available threat categories. If `code` is provided, it retrieves assessments for species affected by the specified threat(s).

Usage

```
rl_threats(
  code = NULL,
  year_published = NULL,
  latest = NULL,
  possibly_extinct = NULL,
  possibly_extinct_in_the_wild = NULL,
  scope_code = NULL,
  page = 1
)
```

Arguments

`code`	Character. One or more threat codes (e.g., "1"). Use `rl_threats()` to list available threat categories.
`year_published`	Optional. Single or numeric vector of years to filter assessments by publication year.
`latest`	Optional. Logical. If `TRUE`, return only the latest assessment per species.
`possibly_extinct`	Optional. Logical. Filter for species flagged as possibly extinct.
`possibly_extinct_in_the_wild`	Optional. Logical. Filter for species possibly extinct in the wild.
`scope_code`	Optional. Integer One or more scope codes to filter assessments.
`page`	Optional. Integer vector. Specify one or more page numbers to fetch. If `NULL` or `NA`, all pages will be fetched automatically.

Value

A tibble (class `tbl_df`, `tbl`, `data.frame`) where each column represents a unique API response JSON key. If `code = NULL`, the tibble contains available threat categories with columns such as code and description. If `code` is provided, the tibble contains assessment data for the specified threat(s), including threat description, threat code and year.

Examples

```
## Not run:
# List all threat categories
rl_threats()

# Get species affected by agriculture & aquaculture threats (code 2)
rl_threats(code = 2)

# Get species affected by Climate change & severe weather threats published in 2025
rl_threats(
  code = "11",
  year_published = 2025
)

## End(Not run)
```

`rl_use_and_trade` *IUCN Red List use and trade categories*

Description

Retrieve species assessments based on use and trade categories. If `code = NULL`, it returns a list of available use and trade categories. If `code` is provided, it retrieves assessments for species affected by the specified use/trade category(ies).

Usage

```
rl_use_and_trade(
  code = NULL,
  year_published = NULL,
  latest = NULL,
  possibly_extinct = NULL,
  possibly_extinct_in_the_wild = NULL,
  scope_code = NULL,
  page = 1
)
```

Arguments

code	Character. One or more use/trade codes (e.g., "1", "5_2"). Use rl_use_and_trade() to list available categories.
year_published	Optional. Single or numeric vector of years to filter assessments by publication year.
latest	Optional. Logical. If TRUE, return only the latest assessment per species.
possibly_extinct	Optional. Logical. Filter for species flagged as possibly extinct.
possibly_extinct_in_the_wild	Optional. Logical. Filter for species possibly extinct in the wild.
scope_code	Optional. Integer One or more scope codes to filter assessments.
page	Optional. Integer vector. Specify one or more page numbers to fetch. If NULL or NA, all pages will be fetched automatically.

Value

A tibble (class tbl_df, tbl, data.frame) where each column represents a unique API response JSON key. If code = NULL, the tibble contains available use and trade categories with columns such as code and description. If code is provided, the tibble contains assessment data for the specified use/trade category(ies), including description, code, year, latest, and other relevant metatdata.

Examples

```
## Not run:
# List all use and trade categories
rl_use_and_trade()

# Get species used for food - human (code 1)
rl_use_and_trade(code = "1")

# Get species hunted for Sport hunting/specimen collecting published in 2024
rl_use_and_trade(
  code = "15",
  year_published = 2024
)

## End(Not run)
```

---

rl_version *IUCN Red List and API version*

Description

Print the current version of the IUCN Red List of Threatened Species and API

Usage

```
rl_version()
```

Value

Invisibly returns NULL after printing the Red List and API versions.

Examples

```
## Not run:
rl_version()

## End(Not run)
```

Index

CoordinateCleaner::cc_outl(), *8*

rgbif::occ_search(), *28*
rl_aoo, 3
rl_aoo(), *7*, *9*, *15*, *26*, *28*, *48*
rl_assessment_id, 4
rl_biogeographical_realms, 5
rl_biogeographical_realms(), *5*
rl_check_api, 6
rl_check_occurrences, 7
rl_check_occurrences(), *27, 28*
rl_classes, 9
rl_classes(), *9*
rl_comprehensive_groups, 10
rl_comprehensive_groups(), *11*
rl_conservation_actions, 11
rl_conservation_actions(), *12*
rl_countries, 13
rl_countries(), *13*
rl_eoo, 14
rl_eoo(), *4*, *7*, *9*, *26*, *28*, *48*
rl_family, 15
rl_family(), *16*
rl_faos, 17
rl_generation_length, 18
rl_generation_length(), *37, 38*, *48*
rl_green_status, 20
rl_growth_forms, 21
rl_habitats, 22
rl_habitats(), *23*
rl_kingdoms, 23
rl_kingdoms(), *24*
rl_name_resolve, 25
rl_name_resolve(), *27, 28*, *41*
rl_occurrences, 26
rl_occurrences(), *7*, *9*
rl_open_file, 29
rl_orders, 29
rl_orders(), *30*
rl_overall_reduction, 31
rl_overall_reduction(), *38*
rl_phylum, 32
rl_phylum(), *32*
rl_population_trends, 33
rl_population_trends(), *33*
rl_possibly_extinct, 34
rl_possibly_extinct_in_wild, 35
rl_red_list_categories, 36
rl_red_list_categories(), *36*
rl_reduction, 37
rl_reduction(), *31, 32*, *48*
rl_research, 39
rl_research(), *39*
rl_scientific_name, 40
rl_scientific_name(), *25–27*
rl_scopes, 41
rl_scopes(), *42*
rl_set_api, 43
rl_set_api(), *7*
rl_sis, 43
rl_statistics, 44
rl_stresses, 44
rl_stresses(), *45*
rl_systems, 46
rl_systems(), *46*
rl_threat_category, 47
rl_threats, 49
rl_threats(), *49*
rl_use_and_trade, 50
rl_use_and_trade(), *51*
rl_version, 51