Package 'taxify'

November 2, 2026

Title Offline Taxonomic Name Matching Against Darwin Core Backbones

Version 0.7.1

Description Matches taxonomic names against local copies of published backbones ('Catalogue of Life', 'GBIF Backbone Taxonomy', 'ITIS', 'World Flora Online', 'WoRMS', 'NCBI Taxonomy', 'Open Tree of Life', 'Euro+Med', 'Species Fungorum', 'AlgaeBase', 'FishBase', 'SeaLifeBase', 'Reptile Database', 'LCVP', 'WCVP', 'Mammal Diversity Database', 'AviList', 'LPSN') and two vegetation classifications, and resolves each name to its accepted name. It performs the name-resolution step of 'taxize' and 'WorldFlora', run offline against these files. Names are cleaned of authors and qualifiers, matched exactly and then by edit distance within their genus, and followed from synonym to accepted name, with the backbone version recorded on every row. Published trait, status and distribution datasets are joined on the accepted name, and so is a table the user supplies. Matching runs in the 'vectra' column-store engine.

License MIT + file LICENSE

URL https://gillescolling.com/taxify/, https://github.com/gcol33/taxify

BugReports https://github.com/gcol33/taxify/issues

Additional_repositories https://gcol33.r-universe.dev

Depends R (>= 4.1.0)

Imports curl, jsonlite, rlang, vectra (>= 0.14.0)

Suggests ape, DBI, knitr, openxlsx2, rgbif, rmarkdown, RSQLite, sf, taxifydb (>= 0.1.19), terra, testthat (>= 3.0.0), TR8, withr

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

Config/roxygen2/version 8.1.0

NeedsCompilation no

Author Gilles Colling [aut, cre, cph] (ORCID:
<https://orcid.org/0000-0003-3070-6066>)

Maintainer Gilles Colling <gilles.colling051@gmail.com>

Repository CRAN

Date/Publication 2026-11-02 13:50:02 UTC

Contents

add_algae_traits . . . 5
add_alien_first_records . . . 7
add_alpine_carabids . . . 8
add_amniote . . . 9
add_amphibio . . . 11
add_anage . . . 12
add_animaltraits . . . 13
add_arctic_traits . . . 14
add_arthropod_traits . . . 15
add_austraits . . . 16
add_avonet . . . 18
add_bacdive . . . 19
add_baseflor . . . 20
add_bee_ostwald . . . 21
add_bet . . . 22
add_betsi_collembola_body_length . . . 24
add_betsi_collembola_traits . . . 25
add_betsi_earthworm_traits . . . 26
add_beukhof . . . 27
add_bien . . . 28
add_birdbase . . . 30
add_blanchard . . . 31
add_brot . . . 32
add_bryoatt . . . 33
add_ccdb . . . 34
add_cefas_btrait . . . 35
add_chelonians . . . 36
add_chowdhury . . . 38
add_classification . . . 39
add_clopla . . . 40
add_col_info . . . 41
add_combine . . . 42
add_combine_imputed . . . 43
add_combine_reported . . . 45
add_common_names . . . 46
add_copepod_traits . . . 47
add_coral_traits . . . 48
add_data . . . 50
add_diaz_traits . . . 52

add_disperse . . . 53
add_eberswalde . . . 54
add_ecoflora . . . 55
add_ecomorphosis . . . 57
add_edwards_phyto . . . 58
add_eive . . . 59
add_ellers_collembola . . . 60
add_elton_traits . . . 61
add_epa_freshwater . . . 63
add_eupolltrait . . . 64
add_eurobat . . . 65
add_euromed_distribution . . . 66
add_faprotax . . . 67
add_finand . . . 68
add_fishbase . . . 69
add_fishmorph . . . 71
add_fishtraits . . . 72
add_floraweb . . . 73
add_freshwater_insects_conus . . . 75
add_frugivoria . . . 76
add_fungalroot . . . 77
add_fungal_traits . . . 78
add_funguild . . . 79
add_gbif_info . . . 81
add_gidias . . . 82
add_gift . . . 84
add_globi . . . 85
add_globtherm . . . 86
add_glonaf . . . 87
add_gmpd . . . 88
add_griis . . . 90
add_groot . . . 91
add_gwdd . . . 92
add_homerange . . . 93
add_hosts . . . 94
add_huang_amph . . . 95
add_hybrid_info . . . 96
add_hydraulics . . . 97
add_imageomics_neon . . . 98
add_inrae_collembola_traits . . . 99
add_invacost . . . 100
add_italic . . . 101
add_iucn . . . 103
add_kew_cvalues . . . 104
add_kew_sid . . . 105
add_leda . . . 106
add_leptraits . . . 107
add_madin . . . 109

add_monograph_collembola_body_length . . . 110
add_nesttrait . . . 111
add_noddb . . . 112
add_nztd . . . 113
add_octocoral . . . 114
add_odonata . . . 115
add_pantheria . . . 116
add_parravicini . . . 117
add_pelagic . . . 118
add_phylacine . . . 119
add_pignatti . . . 120
add_plantatt . . . 121
add_plazi_collembola_body_length . . . 122
add_pottier . . . 124
add_quimbayo . . . 125
add_ramond . . . 126
add_repttraits . . . 127
add_rimet_phyto . . . 128
add_saproxylic . . . 129
add_sealifebase . . . 130
add_sharkipedia . . . 132
add_sheld . . . 133
add_spider_traits . . . 134
add_sworm . . . 135
add_tetradensity . . . 136
add_thermofresh . . . 137
add_trait . . . 138
add_tree_of_sex . . . 141
add_usda_fungus_host . . . 142
add_useful_plants . . . 143
add_virion . . . 144
add_wcvp . . . 145
add_wfo_info . . . 146
add_zanne . . . 147
add_zooplankton . . . 148
children . . . 149
cite . . . 150
class2tree . . . 151
comm2sci . . . 152
downstream . . . 154
enrichment_cols . . . 155
enrichment_groups . . . 156
euromed_areas . . . 157
export_data . . . 157
gbif_backmatch . . . 158
gbif_fetch . . . 159
gbif_request . . . 160
gift_traits . . . 162

id2name . . . 163
inspect . . . 164
install_backbones . . . 166
list_backbones . . . 167
list_enrichments . . . 168
list_traits . . . 169
lookup_genus . . . 170
lowest_common . . . 170
parse_name . . . 171
reconcile . . . 173
sci2comm . . . 174
synonyms . . . 176
taxify . . . 177
taxify_build . . . 183
taxify_build_register . . . 183
taxify_clean . . . 184
taxify_clear_cache . . . 185
taxify_databases . . . 186
taxify_data_dir . . . 187
taxify_download . . . 187
taxify_download_enrichment . . . 188
taxify_example_data . . . 190
taxify_ids . . . 190
taxify_load_register . . . 192
taxify_lock . . . 193
taxify_long . . . 194
taxify_pin . . . 195
taxify_refresh_manifest . . . 196
taxify_regions . . . 197
taxify_register_coverage . . . 198
taxify_restore . . . 198
taxify_store . . . 199
trait_info . . . 200
upstream . . . 201

Index **203**

`add_algae_traits` *Add macroalgal functional traits (AlgaeTraits)*

Description

Joins AlgaeTraits (Vranken et al. 2023) macroalgal functional traits to a `taxify()` result by looking up `accepted_name`. AlgaeTraits provides morphological, ecological, and life-history traits for European seaweeds.

Usage

```
add_algae_traits(x, cols = NULL, verbose = TRUE)
```

Arguments

x	A data.frame returned by taxify().
cols	Which columns to attach: NULL (default) the curated set, "all" every column the source carries, or a character vector of names. See enrichment_cols.
verbose	Logical. Default TRUE.

Details

Source: AlgaeTraits (Vranken et al. 2023, VLIZ Marine Data Archive, CC BY 4.0). Coverage: ~1,745 European macroalgae species.

Value

The same data.frame with additional columns:

algae_body_size_cm Maximum body size in centimetres.

algae_growth_form Growth form / body shape (e.g., filamentous, foliose, crustose).

algae_calcification Calcification type (e.g., uncalcified, articulated, encrusting).

algae_life_span Life span category (annual, perennial, etc.).

algae_tidal_zone Tidal zonation (e.g., supralittoral, eulittoral, sublittoral).

algae_wave_exposure Wave exposure tolerance (sheltered, moderately exposed, exposed).

algae_environment Habitat environment (marine, brackish, freshwater).

algae_substrate Environmental position / substrate type.

References

Vranken S et al. (2023) AlgaeTraits: a trait database for (European) seaweeds. Earth System Science Data 15:2711-2754. doi:10.5194/essd-15-2711-2023

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify("Fucus vesiculosus", backbone = "gbif") |>
  add_algae_traits()

options(old)
```

`add_alien_first_records` *Add alien species first record years*

Description

Joins alien species first record data to a `taxify()` result, filtered by location. Data from the FirstRecords dataset (Seebens & Renard Truong 2026; Seebens et al. 2017).

Usage

```
add_alien_first_records(x, location, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`location`	Character. Location key(s), or `"all"`. A location that is a whole country is given by its ISO 3166-1 alpha-2 code (e.g. `"AT"`, `"US"`); an island or other part of a country that the source records as a region of its own is given by its name (e.g. `"Hawaii"`, `"Canary Islands"`). • Single location (e.g., `"AT"`): adds columns without suffix. • Multiple locations (e.g., `c("AT", "DE")`): adds columns with a location suffix (e.g., `alien_first_record_AT`). • `"all"`: adds one column set per location in the dataset. List the locations covered with `enrichment_groups("alien_first_records")`.
`cols`	Character vector of value columns to attach (from `alien_first_record`, `alien_first_record_source`, `alien_first_record_reference`, plus the dataset's further columns such as `country_code` and `alien_first_record_status`), or `"all"`. `NULL` (default) attaches the first three.
`verbose`	Logical. Default `TRUE`.

Details

The source's locations do not overlap: a country's location excludes the islands and other parts recorded separately, so `location = "US"` is the first record for the United States without Hawaii and Alaska, and a species first recorded on Hawaii has no `"US"` record unless the source holds one for the country itself. Seebens et al. (2017) define the set as "282 non-overlapping regions (countries and sub-national regions such as islands)". Every row carries `country_code`, the ISO 3166-1 alpha-2 code of the country the location lies in, so a country-wide first record is an explicit roll-up: request the country's locations and take the earliest year.

Source: FirstRecords v4.0 (Seebens & Renard Truong 2026, doi:10.5281/zenodo.18759840), CC BY 4.0, one row per species x location.

Value

The same data.frame with additional column(s):

alien_first_record Year of the first record (integer), or NA if not recorded for that location.

alien_first_record_source Database that contributed the record (e.g., "GAVIA", "CABI ISC").

alien_first_record_reference Original citation or reference for the record.

References

Seebens H, Blackburn TM, Dyer EE, et al. (2017) No saturation in the accumulation of alien species worldwide. Nature Communications 8:14435.

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify("Robinia pseudoacacia") |>
  add_alien_first_records(location = "AT")

taxify(c("Robinia pseudoacacia", "Ailanthus altissima")) |>
  add_alien_first_records(location = c("AT", "DE", "Hawaii"))

options(old)
```

---

add_alpine_carabids *Add Alpine ground-beetle traits (Chamberlain et al.)*

Description

Joins body size and wing morphology for the carabid fauna of the Italian Alps to a taxify() result by accepted name.

Usage

```
add_alpine_carabids(x, cols = NULL, verbose = TRUE)
```

Arguments

x	A data.frame returned by taxify().
cols	Which columns to attach: NULL (default) both, or a character vector of names. See enrichment_cols.
verbose	Logical. Default TRUE.

Details

Source: Chamberlain et al. (2020), 185 species recorded at 416 sites between 697 and 2840 m in the Italian Alps. CC0.

The widest European carabid trait table after `add_chowdhury()`, and the one that reaches the Alpine fauna the lowland compilations miss. Like every European carabid source its body sizes trace back to carabids.org, so agreement with those is not corroboration; it runs 1.03 against them, the same quantity without being the verbatim copy `add_eberswalde()` is.

`wing_morph` arrives as bare letters with no legend in the file. The codes were read off the data rather than assumed: crossed against `add_chowdhury()`'s words, `b` is short-winged, `m` long-winged and `d` dimorphic.

Value

The same data.frame with `alpine_` columns: numeric `body_length_mm`, categorical `wing_morph`.

References

Chamberlain D, Gobbi M, Negro M, et al. (2020) Trait-modulated decline of carabid beetle occurrence along elevation gradients across the European Alps. Journal of Biogeography 47:1030-1041. doi:10.1111/jbi.13792

Data: doi:10.5061/dryad.fn2z34tq1

Examples

```
## Not run:
taxify(c("Abax exaratus", "Carabus depressus")) |>
  add_alpine_carabids()

## End(Not run)
```

`add_amniote` *Add amniote life-history traits (Amniote Life History Database)*

Description

Joins uniform life-history traits for birds, mammals and reptiles to a `taxify()` result by looking up `accepted_name`.

Usage

```
add_amniote(x, cols = NULL, verbose = TRUE)
```

Arguments

x	A data.frame returned by taxify().
cols	Which columns to attach: NULL (default) the curated set, "all" every column the source carries, or a character vector of names. See enrichment_cols.
verbose	Logical. Default TRUE.

Details

Source: Amniote Life History Database (Myhrvold et al. 2015, Ecology, CC0). Coverage: 21,322 species across birds, mammals and reptiles.

Value

The same data.frame with additional columns:

amniote_class Taxonomic class (Aves/Mammalia/Reptilia).

amniote_adult_body_mass_g Adult body mass (g).

amniote_no_sex_body_mass_g Unsexed adult body mass (g).

amniote_female_body_mass_g Female body mass (g).

amniote_male_body_mass_g Male body mass (g).

amniote_adult_svl_cm Adult snout-vent length (cm).

amniote_maximum_longevity_y Maximum longevity (years).

amniote_litter_clutch_size Litter or clutch size (count).

amniote_clutches_per_y Litters or clutches per year (count).

amniote_egg_mass_g Egg mass (g).

amniote_incubation_d Incubation period (days).

amniote_female_maturity_d Age at female maturity (days).

amniote_gestation_d Gestation length (days).

amniote_weaning_d Weaning age (days).

amniote_birth_hatching_wt_g Birth or hatching weight (g).

References

Myhrvold NP et al. (2015) An amniote life-history database to perform comparative analyses with birds, mammals, and reptiles. Ecology 96:3109. doi:10.1890/150846R.1

Examples

```
## Not run:
taxify("Accipiter badius", backbone = "gbif") |>
  add_amniote()

## End(Not run)
```

add_amphibio *Add amphibian life-history traits (AmphiBIO)*

Description

Joins AmphiBIO amphibian life-history and ecological traits to a `taxify()` result by looking up `accepted_name`.

Usage

```
add_amphibio(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: AmphiBIO (Oliveira et al. 2017, CC BY 4.0). Coverage: ~6,800 amphibian species. Amphibians only.

Value

The same data.frame with additional columns:

body_size_mm Maximum body size in mm (snout-vent length).

age_maturity_y Age at maturity in years.

longevity_yr Maximum longevity in years.

litter_size Clutch/litter size.

reproductive_output Reproductive output per year.

offspring_size_mm Offspring size in mm.

direct_development Direct development (0/1).

larval Has larval stage (0/1).

aquatic Aquatic habitat (0/1).

fossorial Fossorial habitat (0/1).

arboreal Arboreal habitat (0/1).

diurnal Diurnal activity (0/1).

nocturnal_amphibio Nocturnal activity (0/1). Named `nocturnal_amphibio` to avoid collision with EltonTraits' `nocturnal` column.

References

Oliveira BF, Sao-Pedro VA, Santos-Barrera G, Penone C, Costa GC (2017) AmphiBIO, a global database for amphibian ecological traits. Scientific Data 4:170123.

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify("Bufo bufo", backbone = "gbif") |>
  add_amphibio()

options(old)
```

---

add_anage *Add longevity and life-history traits (AnAge)*

Description

Joins AnAge (Animal Ageing and Longevity Database) traits to a `taxify()` result by looking up `accepted_name`.

Usage

```
add_anage(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: AnAge (de Magalhaes & Costa 2009, CC BY). Coverage: ~4.7k vertebrate species (mammals, birds, reptiles, amphibians, fish).

Value

The same data.frame with additional columns:

max_longevity_yr Maximum longevity in years.

anage_body_mass_g Adult body mass in grams.

metabolic_rate_w Basal metabolic rate in watts.

female_maturity_d Female age at sexual maturity in days.

male_maturity_d Male age at sexual maturity in days.

gestation_incubation_d Gestation or incubation length in days.

anage_litter_size Litter or clutch size.

birth_mass_g Mass at birth in grams.

growth_rate Growth rate (1/days).

temperature_k Body temperature in Kelvin.

References

de Magalhaes JP, Costa J (2009) A database of vertebrate longevity records and their relation to other life-history traits. Journal of Evolutionary Biology 22:1770-1774.

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify("Vulpes vulpes", backbone = "gbif") |>
  add_anage()

options(old)
```

`add_animaltraits` *Add cross-taxon body mass and metabolic rate (AnimalTraits)*

Description

Joins AnimalTraits body mass and metabolic rate data to a `taxify()` result by looking up `accepted_name`.

Usage

```
add_animaltraits(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: AnimalTraits (Herberstein et al. 2022, CC0). Coverage: ~2k species across arthropods, vertebrates, molluscs, and annelids. Individual-level observations aggregated to species medians.

Value

The same data.frame with additional columns:

animaltraits_body_mass_kg Median body mass in kg.

animaltraits_metabolic_rate_w Median metabolic rate in watts.

References

Herberstein ME et al. (2022) AnimalTraits: a curated animal trait database for body mass, metabolic rate and brain size. Scientific Data 9:265.

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify("Drosophila melanogaster", backbone = "gbif") |>
  add_animaltraits()

options(old)
```

---

add_arctic_traits *Add Arctic marine benthos traits*

Description

Joins Arctic Traits Database functional traits to a `taxify()` result by `accepted_name`. Each fuzzy-coded trait is reduced to its dominant category.

Usage

```
add_arctic_traits(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Arctic Traits Database (Degen & Faulwetter 2019, University of Vienna PHAIDRA, CC-BY 4.0).

Value

The same data.frame with categorical `arctic_traits_` columns: `feeding_habit`, `skeleton`, `reproduction`, `larval_development`, `size`, `living_habit`, `body_form`, `mobility`, `bioturbation`, `depth_range`, `trophic_level`, `fragility`, `sociability`, `longevity`.

References

Degen R, Faulwetter S (2019) The Arctic Traits Database. University of Vienna. doi:10.25365/phaidra.49

Examples

```
## Not run:
taxify("Astarte borealis", backbone = "gbif") |>
  add_arctic_traits()

## End(Not run)
```

`add_arthropod_traits` *Add arthropod life-history traits (NW European Arthropods)*

Description

Joins the Northwestern European Arthropod Life Histories dataset to a `taxify()` result by looking up `accepted_name`.

Usage

```
add_arthropod_traits(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Logghe et al. (2025, CC BY-NC). Coverage: ~4.9k arthropod species from NW Europe across 10 orders (Coleoptera, Hemiptera, Orthoptera, Araneae, Diptera, Hymenoptera, Lepidoptera, etc.).

Value

The same data.frame with additional columns:

arthropod_body_size_mm Body size in mm.

arthropod_dispersal Dispersal ability (0–1 ratio within order).

arthropod_voltinism Mean number of generations per year.

arthropod_fecundity Fecundity (number of eggs/offspring).

arthropod_development_d Development time in days.

arthropod_lifespan_d Adult lifespan in days.

arthropod_thermal_mean Mean thermal niche (degrees C).

arthropod_diurnality Activity period (diurnal/nocturnal/both).

arthropod_feeding_guild Feeding guild of adult.

arthropod_trophic_range Trophic range of adult (specialist/generalist).

References

Logghe A et al. (2025) An in-depth dataset of northwestern European arthropod life histories and ecological traits. Biodiversity Data Journal 13:e146785.

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify("Abax parallelepipedus", backbone = "gbif") |>
  add_arthropod_traits()

options(old)
```

`add_austraits` *Add Australian plant traits (AusTraits)*

Description

Joins species-level plant functional traits to a `taxify()` result by looking up `accepted_name`. Values are aggregated from the long-format AusTraits database (numeric traits by median, categorical traits by mode).

Usage

```
add_austraits(x, cols = NULL, verbose = TRUE)
```

Arguments

x	A data.frame returned by taxify().
cols	Which columns to attach: NULL (default) the curated set, "all" every column the source carries, or a character vector of names. See enrichment_cols.
verbose	Logical. Default TRUE.

Details

Source: AusTraits (Falster et al. 2021, Scientific Data, CC BY 4.0). Coverage: ~33k Australian plant taxa.

Value

The same data.frame with additional columns:

austraits_plant_growth_form Plant growth form.

austraits_life_history Life history (annual/perennial/...).

austraits_woodiness Woodiness.

austraits_photosynthetic_pathway Photosynthetic pathway (C3/C4/CAM).

austraits_dispersal_syndrome Dispersal syndrome.

austraits_resprouting_capacity Resprouting capacity (fire response).

austraits_flowering_time Flowering time.

austraits_plant_height_m Plant height (m).

austraits_leaf_length_mm Leaf length (mm).

austraits_leaf_width_mm Leaf width (mm).

austraits_leaf_area_mm2 Leaf area (mm2).

austraits_leaf_mass_per_area Leaf mass per area (g/m2; SLA is its reciprocal).

austraits_leaf_n_per_dry_mass Leaf nitrogen per dry mass (mg/g).

austraits_leaf_p_per_dry_mass Leaf phosphorus per dry mass (mg/g).

austraits_seed_dry_mass_mg Seed dry mass (mg).

austraits_wood_density_g_cm3 Wood density (g/cm3).

References

Falster D et al. (2021) AusTraits, a curated plant trait database for the Australian flora. Scientific Data 8:254. doi:10.1038/s41597021010066

Examples

```
old <- options(taxify.data_dir = taxify_example_data())

taxify("Eucalyptus globulus", backbone = "gbif") |>
  add_austraits()

options(old)
```

`add_avonet` *Add bird morphology and migration (AVONET)*

Description

Joins AVONET species-level averages for bird morphology, ecology, and migration to a `taxify()` result by looking up `accepted_name`.

Usage

```
add_avonet(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach. `NULL` (the default) attaches the curated set; a character vector of column names attaches just those, and `"all"` attaches every column the source carries (see `enrichment_cols()`).
`verbose`	Logical. Default `TRUE`.

Details

Source: AVONET (Tobias et al. 2022, Figshare, CC BY 4.0). Coverage: ~11k bird species. Birds only.

Value

The same data.frame with additional columns:

beak_length Beak length in mm (culmen, species mean).

beak_depth Beak depth in mm (species mean).

wing_length Wing length in mm (species mean).

tail_length Tail length in mm (species mean).

tarsus_length Tarsus length in mm (species mean).

avonet_body_mass_g Body mass in grams (species mean).

hand_wing_index Hand-wing index (pointedness, species mean).

habitat Primary habitat classification.

trophic_level Trophic level classification.

trophic_niche Trophic niche classification.

migration Migration strategy: `"sedentary"`, `"partial"`, or `"full"`.

References

Tobias JA et al. (2022) AVONET: morphological, ecological and geographical data for all birds. Ecology Letters 25:581-597.

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify("Parus major", backbone = "gbif") |>
  add_avonet()

options(old)
```

`add_bacdive` *Add bacterial and archaeal strain phenotypes (BacDive)*

Description

Joins per-species microbial phenotype and growth-condition traits from BacDive, the Bacterial Diversity Metadatabase (DSMZ), to a `taxify()` result by looking up `accepted_name`. Strain-level records are aggregated to one row per species (categorical traits by mode, numeric by median); temperature and pH prefer the optimum measurement, falling back to the growth measurement.

Usage

```
add_bacdive(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: BacDive (Reimer et al.), DSMZ, CC BY 4.0. ~18.6k bacterial and archaeal species with at least one phenotypic trait.

Value

The same data.frame with additional columns:

gram_stain Gram reaction (positive / negative / variable).

cell_shape Cell morphology (rod, coccus, ...).

motility motile / non-motile.

oxygen_metabolism Oxygen tolerance (aerobe, anaerobe, facultative anaerobe, microaerophile, ...).

cell_length_um, cell_width_um Cell dimensions in micrometres.

optimal_growth_temp_c Optimal (or reported growth) temperature, C.

optimal_growth_ph Optimal (or reported growth) pH.

References

Reimer LC et al. (2022) BacDive in 2022: the knowledge base for standardized bacterial and archaeal data. Nucleic Acids Research 50:D741-D746. doi:10.1093/nar/gkab961

Examples

```
## Not run:
taxify("Escherichia coli", backbone = "gbif") |>
  add_bacdive()

## End(Not run)
```

`add_baseflor` *Add plant traits from Baseflor (Catminat / Julve)*

Description

Joins Baseflor (Julve, Programme Catminat) plant traits to a `taxify()` result by looking up `accepted_name`. Baseflor covers the vascular flora of France and neighbouring regions, providing flowering phenology, pollination and breeding biology, dispersal mode, and floral and fruit morphology.

Usage

```
add_baseflor(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Baseflor, Programme Catminat (Julve 1998 ff.). Coverage: ~7,000 vascular plant taxa of France and neighbouring regions. Data are released under ODbL 1.0 / CC BY-SA 2.0.

For ecological indicator values on the light, temperature, moisture, reaction, and nutrient axes, see `add_eive()` (European calibration). For Raunkiaer life form and seed, leaf, and clonality traits of the Northwest European flora, see `add_leda()`.

Value

The same data.frame with additional columns:

flower_begin_month First month of flowering (1-12).

flower_end_month Last month of flowering (1-12). A value smaller than `flower_begin_month` denotes a flowering period that wraps across the new year (e.g. begin 10, end 6).

pollination_vector Pollination vector(s): insect, wind, water, self, apogamy. Comma-separated when more than one applies.

dispersal_mode Diaspore dispersal mode(s): anemochory, barochory, epizoochory, endozoochory, myrmecochory, hydrochory, autochory, dyszoochory. Comma-separated when more than one applies.

breeding_system Sexual system: hermaphroditic, monoecious, dioecious, gynodioecious, androdioecious, gynomonoecious, polygamous.

flower_colour Flower colour(s): white, yellow, pink, green, blue, brown, black. Comma-separated when more than one applies.

fruit_type Fruit type: achene, capsule, caryopsis, drupe, legume, silique, berry, follicle, cone, samara, pyxid.

woody_growth_form Woody growth form for woody taxa: tree, small tree, large tree, shrub, bush, subshrub, liana, parasite. NA for non-woody (herbaceous) taxa.

continentality Ellenberg-style continentality indicator value (1-9), the axis absent from EIVE.

salinity Ellenberg-style salinity indicator value (0-9), the axis absent from EIVE.

References

Julve, Ph. (1998 ff.) baseflor. Index botanique, ecologique et chorologique de la Flore de France. Programme Catminat.

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify("Bellis perennis") |>
  add_baseflor()

options(old)
```

`add_bee_ostwald` *Add bee morphometrics (Ostwald)*

Description

Joins global bee morphological traits to a `taxify()` result by `accepted_name`. Long-format measurements are reduced to species medians.

Usage

```
add_bee_ostwald(x, cols = NULL, verbose = TRUE)
```

Arguments

x	A data.frame returned by taxify().
cols	Which columns to attach: NULL (default) the curated set, "all" every column the source carries, or a character vector of names. See enrichment_cols.
verbose	Logical. Default TRUE.

Details

Source: Ostwald et al. global bee morphology (Zenodo, CC-BY 4.0).

Value

The same data.frame with numeric columns bee_ostwald_itd_mm (intertegular distance), bee_ostwald_forewing_length_ bee_ostwald_tongue_length_mm, bee_ostwald_tongue_width_mm, bee_ostwald_body_length_mm, bee_ostwald_thorax_length_mm, bee_ostwald_hair_length_mm, bee_ostwald_hair_coverage_pct.

References

Ostwald MM et al. (2024) A global database of bee morphological traits. Zenodo. doi:10.5281/zenodo.13366989

Examples

```
## Not run:
taxify("Apis mellifera", backbone = "gbif") |>
  add_bee_ostwald()

## End(Not run)
```

---

add_bet *Add bryophyte traits (Bryophytes of Europe Traits)*

Description

Joins species-level bryophyte traits to a taxify() result by looking up accepted_name.

Usage

```
add_bet(x, cols = NULL, verbose = TRUE)
```

Arguments

x	A data.frame returned by taxify().
cols	Which columns to attach: NULL (default) the curated set, "all" every column the source carries, or a character vector of names. See enrichment_cols.
verbose	Logical. Default TRUE.

Details

Source: Bryophytes of Europe Traits (van Zuijlen et al. 2023, EnviDat, CC BY-SA 4.0). Coverage: ~1.8k bryophyte species.

Value

The same data.frame with additional columns:

bet_growth_form Growth form (acrocarpous/pleurocarpous/thalloid/...).

bet_life_form Life form (cushion/mat/turf/weft/...).

bet_life_strategy Life strategy (During).

bet_sexual_condition Sexual condition (monoicous/dioicous).

bet_shoot_size_mm Mean shoot size (mm).

bet_generation_length_y Generation length (years).

bet_spore_diameter_um Mean spore diameter (micrometres).

bet_ind_light Ellenberg light indicator value.

bet_ind_temperature Ellenberg temperature indicator value.

bet_ind_moisture Ellenberg moisture indicator value.

bet_ind_reaction_ph Ellenberg reaction (pH) indicator value.

bet_ind_nitrogen Ellenberg nitrogen indicator value.

bet_substrate_soil Occurs on soil (0/1).

bet_substrate_rock Occurs on rock (0/1).

bet_substrate_bark Occurs on bark (0/1).

bet_substrate_deadwood Occurs on deadwood (0/1).

bet_substrate Every substrate class the species occurs on, as a pipe-delimited set over soil, rock, bark, wood (deadwood), living_plants (epiphytic on leaves or other bryophytes) and dung_carcass, e.g. "soil|rock|bark". add_trait() "substrate" reduces it to one primary class.

bet_epiphyte Epiphytic (0/1).

bet_redlist_category IUCN European Red List category.

References

van Zuijlen K et al. (2023) Bryophytes of Europe Traits (BET): a fundamental dataset for European bryophyte ecology. EnviDat. doi:10.16904/envidat.348

Examples

```
## Not run:
taxify("Abietinella abietina", backbone = "gbif") |>
  add_bet()

## End(Not run)
```

`add_betsi_collembola_body_length` *Add Collembola body length (BETSI export)*

Description

Joins per-species springtail body length to a `taxify()` result by looking up `accepted_name`. The value is the median over the BETSI compilation's per-source measurements, with the observed range and the measurement and source counts behind it.

Usage

```
add_betsi_collembola_body_length(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) all, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: body length values from the BETSI database (Biological and Ecological Traits of Soil Invertebrates), the 2017 all-Collembola export requested by Bonfanti (2018), CC BY-NC 4.0. Coverage: 1,374 species. The European-compendia body-length floor: it agrees with the Ellers et al. body size at Pearson r = 0.96 over 262 shared species and with the Plazi treatment values at r = 0.906. Body length is also available through `add_trait()("body_length")`, which coalesces it with the Plazi, monograph and other springtail sources.

Value

The same data.frame with additional columns:

betsi_body_length_mm Body length (mm), median of the per-source measurements.

betsi_body_length_min_mm Smallest measurement (mm).

betsi_body_length_max_mm Largest measurement (mm).

betsi_body_length_n Number of measurements aggregated.

betsi_body_length_sources Number of distinct literature sources.

References

Bonfanti J (2018) Body length trait values from the BETSI database, on all Collembola species. Zenodo. doi:10.5281/zenodo.1292461

Examples

```
## Not run:
taxify("Isotoma viridis", backbone = "gbif") |>
  add_betsi_collembola_body_length()

## End(Not run)
```

`add_betsi_collembola_traits` *Add Collembola traits (Lu et al. 2025)*

Description

Joins per-species springtail functional traits to a `taxify()` result by looking up `accepted_name`. Output columns are prefixed `betsi_ct_`.

Usage

```
add_betsi_collembola_traits(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: per-species Collembola traits from Lu et al. (2025), Appendix S1. The morphological traits (body size, antenna/body ratio, furca, pigmentation, ocelli, reproduction) are derived from the BETSI database (Pey et al. 2014); vertical stratification and trophic position were measured in that study and the life form assigned after Potapov et al. (2016). Coverage: 26 species. Body length is also available through `add_trait()("body_length")`.

Value

The same data.frame with additional columns:

betsi_ct_body_length_mm Maximum body length (mm).

betsi_ct_antenna_body_ratio Antenna to body length ratio.

betsi_ct_ocelli_number Number of ocelli.

betsi_ct_furca Furca (springing organ) development.

betsi_ct_pigment_scaled Pigmentation (scaled).

betsi_ct_reproduction Reproduction mode.

betsi_ct_stratification_scaled Vertical stratification (scaled).

betsi_ct_trophic_position Trophic position.

betsi_ct_life_form Life form (after Potapov et al. 2016).

References

Lu J-Z et al. (2025) Mixed forests with native species mitigate impacts of introduced Douglas fir on soil decomposers (Collembola). Ecological Applications 35:e70034. doi:10.1002/eap.70034

Examples

```
## Not run:
taxify("Isotoma viridis", backbone = "gbif") |>
  add_betsi_collembola_traits()

## End(Not run)
```

`add_betsi_earthworm_traits`

Add earthworm traits (Pelosi et al. 2014)

Description

Joins fuzzy-coded earthworm functional traits to a `taxify()` result by looking up `accepted_name`. Output columns are prefixed `betsi_ew_`.

Usage

```
add_betsi_earthworm_traits(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) all, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: earthworm functional traits compiled from the BETSI database (Biological and Ecological Traits of Soil Invertebrates) and published in Pelosi et al. (2014), Appendix 1. Coverage: 11 species. Earthworm body length and soil carbon preference are given as fuzzy affinity vectors rather than a single value, so they are not passed to the scalar `add_trait()` verb.

Value

The same data.frame with additional columns, one per modality bin of seven fuzzy-coded traits: body length, body mass to length ratio, cocoon diameter, epithelium, typhlosolis, soil carbon preference and vertical distribution. Each trait is fuzzy coded, so a species' affinities across the bins of one trait (columns sharing a `betsi_ew_<trait>__` stem) sum to 100. See `enrichment_cols` for the full column list.

References

Pelosi C, Pey B, Hedde M, et al. (2014) Reducing tillage in cultivated fields increases earthworm functional diversity. Applied Soil Ecology 83:79-87. doi:10.1016/j.apsoil.2013.10.005

Examples

```
## Not run:
taxify("Lumbricus terrestris", backbone = "gbif") |>
  add_betsi_earthworm_traits()

## End(Not run)
```

`add_beukhof` *Add marine fish traits (Beukhof)*

Description

Joins North Atlantic / NE Pacific shelf marine-fish life-history and ecology traits to a `taxify()` result by `accepted_name` (species-level summaries).

Usage

```
add_beukhof(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Beukhof et al. (2019) marine fish trait collection (PANGAEA, CC-BY 4.0).

Value

The same data.frame with `beukhof_` columns: numeric `trophic_level`, `aspect_ratio`, `offspring_size`, `age_maturity`, `fecundity`, `length_infinity_cm`, `growth_coefficient`, `length_max_cm`; categorical `habitat`, `feeding_mode`, `body_shape`, `fin_shape`, `spawning_type`.

References

Beukhof E et al. (2019) A trait collection of marine fish species from North Atlantic and Northeast Pacific continental shelf seas. PANGAEA. doi:10.1594/PANGAEA.900866

Examples

```
## Not run:
taxify("Gadus morhua", backbone = "gbif") |>
  add_beukhof()

## End(Not run)
```

`add_bien` *Add plant traits (BIEN)*

Description

Joins BIEN plant functional traits to a `taxify()` result by looking up `accepted_name`. Values are species-level aggregates of public-access BIEN records (numeric by median, categorical by mode).

Usage

```
add_bien(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: BIEN (Botanical Information and Ecology Network; Maitner et al. 2018, Methods Ecol Evol, CC BY). Coverage: tens of thousands of vascular plant species.

Value

The same data.frame with additional columns:

bien_plant_height_m Whole-plant height (m).

bien_max_plant_height_m Maximum whole-plant height (m).

bien_dbh_cm Diameter at breast height (cm).

bien_sla_mm2_mg Leaf area per leaf dry mass (SLA, mm2/mg).

bien_leaf_area_mm2 Leaf area (mm2).

bien_leaf_dry_mass_mg Leaf dry mass (mg; BIEN records g).

bien_leaf_n_per_dry_mass Leaf nitrogen per dry mass (mg/g).

bien_leaf_p_per_dry_mass Leaf phosphorus per dry mass (mg/g).

bien_leaf_thickness_mm Leaf thickness (mm).

bien_seed_mass_mg Seed mass (mg).

bien_wood_density_g_cm3 Stem wood density (g/cm3).

bien_leaf_lifespan Leaf life span (months).

bien_growth_form Whole-plant growth form.

bien_woodiness Whole-plant woodiness.

bien_dispersal_syndrome Whole-plant dispersal syndrome.

bien_flower_color Flower colour.

References

Maitner BS et al. (2018) The BIEN R package: A tool to access the Botanical Information and Ecology Network (BIEN) database. Methods in Ecology and Evolution 9:373-379. doi:10.1111/2041210X.12861

Examples

```
old <- options(taxify.data_dir = taxify_example_data())

taxify("Quercus alba", backbone = "gbif") |>
  add_bien()

options(old)
```

`add_birdbase` *Add bird traits (BIRDBASE)*

Description

Joins BIRDBASE biogeography, conservation and life-history traits to a `taxify()` result by looking up `accepted_name`. Traits redundant with `add_avonet()` morphology are not carried.

Usage

```
add_birdbase(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: BIRDBASE (Sekercioglu et al. 2025, figshare, CC BY 4.0). Coverage: ~11.6k bird species.

Value

The same data.frame with additional columns:

birdbase_iucn_status IUCN Red List category.

birdbase_realm Biogeographic realm.

birdbase_latitudinal_zone Latitudinal zone (1 tropical to 5).

birdbase_island_endemic Island-restricted breeding (0/1).

birdbase_restricted_range Restricted-range species (0/1).

birdbase_elevation_min_m Lower elevation limit (m).

birdbase_elevation_max_m Upper elevation limit (m).

birdbase_elevation_range_m Elevational breadth (m).

birdbase_primary_habitat Primary habitat.

birdbase_habitat_breadth Habitat breadth (number of habitats).

birdbase_primary_diet Primary diet.

birdbase_diet_breadth Diet breadth (number of food types).

birdbase_specialization_esi Ecological specialization index.

birdbase_clutch_min Minimum clutch size (eggs).

birdbase_clutch_max Maximum clutch size (eggs).

birdbase_nest_type Nest architecture.

birdbase_flightlessness Volancy (yes/no/partial).

References

Sekercioglu CH et al. (2025) BIRDBASE: a global database of bird ecological and life-history traits. Scientific Data. doi:10.1038/s41597025056153

Examples

```
## Not run:
taxify("Struthio camelus", backbone = "gbif") |>
  add_birdbase()

## End(Not run)
```

`add_blanchard` *Add ant genus defensive traits (Blanchard & Moreau)*

Description

Joins genus-level ant defensive and ecological traits to a `taxify()` result by `genus`.

Usage

```
add_blanchard(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Blanchard & Moreau (2017) ant defensive traits (Dryad, CC0). Joins on genus because the database is genus-resolved.

Value

The same data.frame with `blanchard_` columns: categorical `subfamily`, `spines`, `sting`, `diet`, `nesting`, `foraging`; numeric `colony_size_workers` (joined on genus).

References

Blanchard BD, Moreau CS (2017) Defensive traits in the ant genera database. Dryad. doi:10.5061/dryad.st6sc

Examples

```
## Not run:
taxify("Camponotus pennsylvanicus", backbone = "gbif") |>
  add_blanchard()

## End(Not run)
```

`add_brot` *Add Mediterranean plant traits (BROT 2.0)*

Description

Joins Mediterranean-Basin plant fire-response, regeneration and functional traits to a `taxify()` result by `accepted_name`.

Usage

```
add_brot(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: BROT 2.0 (Tavsanoglu & Pausas 2018, Scientific Data, CC-BY 4.0).

Value

The same data.frame with `brot_` columns: numeric `seed_mass_mg`, `sla_mm2_mg`, `height_m`, `leaf_area_mm2`; categorical `resp_fire`, `growth_form`, `disp_mode`, `fruit_type`, `soil_seed_bank`, `seedling_emergence`.

References

Tavsanoglu C, Pausas JG (2018) A functional trait database for Mediterranean Basin plants (BROT 2.0). Scientific Data 5:180135. doi:10.6084/m9.figshare.c.3843841

Examples

```
old <- options(taxify.data_dir = taxify_example_data())

taxify("Quercus coccifera", backbone = "gbif") |>
  add_brot()

options(old)
```

`add_bryoatt` *Add British and Irish bryophyte attributes (BRYOATT)*

Description

Joins attributes of British and Irish mosses, liverworts and hornworts (Hill et al. 2007) to a `taxify()` result by looking up `accepted_name`. Bryophytes are otherwise almost absent from the bundled trait databases.

Usage

```
add_bryoatt(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

1,194 taxa. The vascular-plant companion is `add_plantatt()`.

This source states no licence (it is copyright the Biological Records Centre), so taxify ships no pre-built copy of it. The first call builds it from the original source on the user's machine, which requires the taxifydb package (`remotes::install_github("gcol33/taxifydb")`). taxify redistributes none of the data. Cite Hill et al. (2007) for the data.

Value

The same data.frame with additional columns. The default set:

bryoatt_ellenberg_light, bryoatt_ellenberg_moisture, bryoatt_ellenberg_reaction, bryoatt_ellenberg_nitrogen, bryoa Ellenberg indicator values calibrated for British and Irish bryophytes.

bryoatt_life_form Life-form code, kept verbatim (`Ts`, `Mr`, `Ms`, ...).

bryoatt_plant_group Plant group: `M` moss, `L` liverwort, `H` hornwort.

bryoatt_status Status code, kept verbatim (`N` native, `AN` alien naturalised, `AR` archaeophyte).

References

Hill MO, Preston CD, Bosanquet SDS, Roy DB (2007) BRYOATT: Attributes of British and Irish Mosses, Liverworts and Hornworts. NERC Centre for Ecology and Hydrology.

Examples

```
## Not run:
# Builds the enrichment on first use (needs taxifydb).
taxify("Polytrichum commune", backbone = "gbif") |>
  add_bryoatt()

## End(Not run)
```

`add_ccdb` *Add plant chromosome numbers (Chromosome Counts Database)*

Description

Joins somatic chromosome numbers from the Chromosome Counts Database (Rice et al. 2015) to a `taxify()` result by looking up `accepted_name`. CCDB aggregates published counts per taxon; records are collapsed to the binomial and reduced to a per-species median with its observed range.

Usage

```
add_ccdb(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: CCDB (ccdb.tau.ac.il), citation requested, no explicit licence, so taxify publishes no pre-built copy. The data is built on the user's machine from the original source, which needs taxifydb installed: `remotes::install_github("gcol33/taxifydb")`.

CCDB's service reports gametic numbers, which taxifydb doubles to the somatic 2n reported here. The spread between `min` and `max` is real ploidy variation among a species' cytotypes rather than measurement error, so a polyploid complex legitimately spans a wide range (*Sedum acre* runs 2n = 40 to 120). Where a species is covered by both, this and the CC BY `add_kew_cvalues()` chromosome column agree exactly for 85% of the 7133 species they share; the rest are compilations differing over which cytotype is typical.

Value

The same data.frame with additional columns. The default set:

ccdb_chromosome_2n Median somatic chromosome number (2n).

ccdb_chromosome_2n_min Lowest 2n reported for the species.

ccdb_chromosome_2n_max Highest 2n reported for the species.

References

Rice A, Glick L, Abadi S, Einhorn M, Kopelman NM, Salman-Minkov A, Mayzel J, Chay O, Mayrose I (2015) The Chromosome Counts Database (CCDB) - a community resource of plant chromosome numbers. New Phytologist 206:19-26.

See Also

`add_kew_cvalues()` for genome size and a CC BY chromosome number.

Examples

```
## Not run:
# Builds the enrichment on first use (requires taxifydb).
taxify("Zea mays", backbone = "gbif") |>
  add_ccdb()

## End(Not run)
```

---

add_cefas_btrait *Add benthic invertebrate traits (Cefas)*

Description

Joins biological traits of North-West European continental-shelf benthic macrofauna from the Cefas biological-traits database to a `taxify()` result by looking up `genus`. The source fuzzy-codes traits at genus level, and each trait is reduced to its highest-scoring modality, so the join is on `genus` rather than `accepted_name`: every species in a coded genus is annotated.

Usage

```
add_cefas_btrait(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: North-West European continental-shelf benthos biological-traits database, Cefas Data Hub (doi:10.14466/CefasDataHub.123), Open Government Licence v3.0.

Value

The same data.frame with additional columns. The default set:

cefas_body_size Maximum body-size class.

cefas_morphology Body morphology.

cefas_lifespan Life-span class.

cefas_living_habit Living habit.

cefas_feeding_mode Feeding mode.

cefas_mobility Mobility.

cefas_bioturbation Bioturbation mode.

`cols = "all"` also attaches egg and larval development and sediment position.

References

Centre for Environment, Fisheries and Aquaculture Science (2022) North-West European continental-shelf benthos biological-traits database. doi:10.14466/CefasDataHub.123

Examples

```
## Not run:
# Downloads the enrichment on first use.
taxify("Abra alba", backbone = "gbif") |>
  add_cefas_btrait()

## End(Not run)
```

`add_chelonians` *Add turtle traits (ChelonianTraits)*

Description

Joins species-level turtle and tortoise traits to a `taxify()` result by looking up `accepted_name`.

Usage

```
add_chelonians(x, cols = NULL, verbose = TRUE)
```

Arguments

x	A data.frame returned by taxify().
cols	Which columns to attach: NULL (default) the curated set, "all" every column the source carries, or a character vector of names. See enrichment_cols.
verbose	Logical. Default TRUE.

Details

Source: ChelonianTraits (Wang et al. 2025, figshare, CC BY 4.0). Coverage: 358 turtle and tortoise species. Numeric values reported as "min-max" ranges in the source are reduced to their midpoint.

Value

The same data.frame with additional columns:

chelonian_carapace_length_mm Maximum straight-line carapace length (mm).

chelonian_max_mass_g Maximum body mass (g).

chelonian_clutch_size_mean Mean clutch size (count).

chelonian_clutch_size_max Maximum clutch size (count).

chelonian_clutches_per_year Clutches per year (count).

chelonian_incubation_d Incubation period (days).

chelonian_age_maturity_y Age at sexual maturity (years).

chelonian_max_lifespan_y Maximum lifespan (years).

chelonian_range_size_km2 Range size (km2).

chelonian_diet Diet (herbivorous/carnivorous/omnivorous).

chelonian_activity_time Activity time.

chelonian_microhabitat Microhabitat (aquatic/terrestrial/...).

chelonian_habitat_type Habitat type.

chelonian_shell_type Shell type (hardshell/softshell).

References

Wang Y et al. (2025) ChelonianTraits: a comprehensive trait database of global turtles and tortoises. figshare. doi:10.6084/m9.figshare.28828241

Examples

```
## Not run:
taxify("Chelonia mydas", backbone = "gbif") |>
  add_chelonians()

## End(Not run)
```

add_chowdhury *Add German ground-beetle traits and occupancy trends (Chowdhury et al. 2025)*

Description

Joins carabid (ground beetle) traits and modelled national occupancy trends to a `taxify()` result by accepted name.

Usage

```
add_chowdhury(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) all of them, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Supporting Information Data S1 of Chowdhury et al. (2025), 383 German carabid species, 382 of them with traits. Ground beetles are one of the most intensively sampled insect groups in ecology, and this is the largest openly reachable block of carabid traits.

`wing_morph` is three-state: long-winged, short-winged, and the 83 dimorphic species that produce both forms. Dimorphism is the trait of interest in carabid dispersal ecology, so it is never collapsed into a flight-capable/flightless binary.

`red_list_germany` is the German national Red List code (`*`, 1, 2, 3, V, R, G, D). `red_list_iucn` restates that same national assessment in IUCN letter codes and is **not** a global IUCN assessment: it maps deterministically onto `red_list_germany` (`*` to LC, 1 to CR, 2 to EN, 3 to VU, V to NT), with no off-diagonal case in all 382 species. A species can be critically endangered at its German range edge and of least concern globally, so these columns stay in this function and out of the global `conservation_status` trait.

`occupancy_trend` is the paper's own result: the modelled change in occupancy probability over two years, negative for a declining species.

Value

The same data.frame with `chowdhury_` columns: numeric `body_length_mm` and `occupancy_trend`; categorical `wing_morph`, `trophic_level`, `habitat_pref`, `red_list_germany`, `red_list_iucn`, `threat_status`, `trend_status`, `trend_significance`.

References

Chowdhury S, Jaureguiberry P, Rada S, et al. (2025) Widespread decline of ground beetles in Germany. Diversity and Distributions. doi:10.1111/ddi.70112

Examples

```
## Not run:
taxify(c("Carabus coriaceus", "Amara aenea")) |>
  add_chowdhury()

## End(Not run)
```

`add_classification` *Add the full higher classification to a taxify result*

Description

Attaches the Linnaean ranks above family (kingdom, phylum, class, order) to a `taxify()` result by joining each matched row back to its backbone. The core `taxify()` output already carries `family` and `genus`; this fills the ranks above them, for whichever ranks the matched backbone stores. Rows matched by different backbones are each joined against their own backbone.

Usage

```
add_classification(
  x,
  ranks = c("kingdom", "phylum", "class", "order"),
  verbose = TRUE
)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`ranks`	Character vector of ranks to attach. Default `c("kingdom", "phylum", "class", "order")`.
`verbose`	Logical. Default `TRUE`.

Value

`x` with the requested rank columns added. A rank a backbone does not store is left `NA` (WFO, for example, carries no ranks above family).

See Also

`add_col_info()`, `add_gbif_info()` for backbone-specific extras.

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify("Naja naja", backbone = "reptiledb") |>
  add_classification()

options(old)
```

`add_clopla` *Add clonal and bud-bank traits (CLO-PLA)*

Description

Joins clonal growth, bud-bank and lifespan traits of the Central European flora (Klimesova et al. 2017) to a `taxify()` result by looking up `accepted_name`. Source records are collapsed to the binomial, so infraspecific rows are aggregated to the species (numeric traits by median, nominal traits by mode).

Usage

```
add_clopla(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every trait the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

2,909 species of the Central European flora.

This source states no licence. It is an Ecological Society of America data paper, free to use with citation, and taxify ships no pre-built copy of it. The first call builds it from the original source on the user's machine, which requires the taxifydb package (`remotes::install_github("gcol33/taxifydb")`). taxify redistributes none of the data. Cite Klimesova et al. (2017) for the data.

Value

The same data.frame with additional columns. The default set:

clopla_clonal Clonal growth present.

clopla_clonalindex Clonal index.

clopla_woody, clopla_annual, clopla_monocarpic, clopla_polycarpic Growth-form and life-cycle flags.

clopla_persistence Persistence of the clonal connection.

clopla_offspring Offspring produced per parent shoot per year.

clopla_spread Lateral spread.

clopla_BBsize, clopla_BBdepth Bud-bank size and depth.

clopla_finalCGO Final clonal growth organ.

`cols = "all"` attaches every trait the source carries (29 in total: the remaining bud-bank counts by depth class, root-derived bud banks, branching, cyclicity and dispersibility), with codes kept verbatim.

References

Klimesova J, Danihelka J, Chrtek J, de Bello F, Herben T (2017) CLO-PLA: a database of clonal and bud-bank traits of the Central European flora. Ecology 98:1179. doi:10.1002/ecy.1745

Examples

```
## Not run:
# Builds the enrichment on first use (needs taxifydb).
taxify("Trifolium repens", backbone = "gbif") |>
  add_clopla()

## End(Not run)
```

---

add_col_info *Add COL-specific columns*

Description

Joins extra Catalogue of Life columns to a `taxify()` result by looking up `taxon_id` in the COL backbone. Only enriches rows where `backbone == "col"`.

Usage

```
add_col_info(x)
```

Arguments

`x`	A data.frame returned by `taxify()` with `backbone == "col"`.

Value

The same data.frame with additional columns:

notho Hybrid type from COL: `"generic"`, `"specific"`, `"infrageneric"`, or `"infraspecific"`.

nomenclaturalCode Nomenclatural code (`"ICN"`, `"ICZN"`, etc.).

nomenclaturalStatus Nomenclatural status.

name_published_in Original publication reference.

kingdom Kingdom classification.

phylum Phylum classification.

col_class Class classification (renamed to avoid conflict with R's `class` function).

order Order classification.

infraspecificEpithet Infraspecific epithet.

is_extinct Logical. Whether the species is extinct (from SpeciesProfile, if available).

is_marine Logical. Whether the species is marine.

is_freshwater Logical. Whether the species is freshwater.

is_terrestrial Logical. Whether the species is terrestrial.

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify("Quercus robur", backbone = "col") |>
  add_col_info()

options(old)
```

`add_combine` *Add mammal traits (COMBINE)*

Description

The default COMBINE function. Attaches the reported (measured or literature compiled) trait values and fills each still-missing cell from COMBINE's phylogenetically imputed table, so a single call reaches the fullest coverage COMBINE offers. A measurement is never overwritten: the reported value wins wherever it exists and the imputed model only fills gaps. Beside every trait sits a `<trait>_src` column recording where that cell came from (`"reported"`, `"imputed"`, or `NA` when neither table has it), so a model estimate is always distinguishable from a measurement.

Usage

```
add_combine(x, cols = NULL, verbose = TRUE)
```

Arguments

x	A data.frame returned by taxify().
cols	Which columns to attach: NULL (default) the curated set, "all" every column the source carries, or a character vector of names. See enrichment_cols.
verbose	Logical. Default TRUE.

Details

For a single-table view use add_combine_reported() (measured values only) or add_combine_imputed() (the imputed table on its own).

Value

The same data.frame with the combine_* trait columns (reported values, gaps filled from the imputed table) and, beside each, a combine_*_src column tagging that cell as "reported", "imputed", or NA. Traits COMBINE does not impute (for example combine_biogeographical_realm) come out "reported" wherever present. If the imputed table is unavailable, the reported values are returned with every _src tag "reported" or NA.

See Also

add_combine_reported(), add_combine_imputed()

Examples

```
## Not run:
# Coverage-filled values with per-trait provenance:
res <- taxify("Osphranter rufus", backbone = "col") |>
  add_combine()
res[, c("combine_gestation_length_d", "combine_gestation_length_d_src")]

## End(Not run)
```

---

add_combine_imputed *Add mammal traits from COMBINE (phylogenetically imputed values)*

Description

Joins the phylogenetically imputed COMBINE table to a taxify() result by looking up accepted_name. This is the companion to add_combine_reported(): the same ~6.2k mammal species, but with the cells the reported table leaves missing filled in by COMBINE's phylogenetic multiple-imputation model, giving near-complete coverage for the life-history traits. Imputed values are model estimates, not measurements, so they are attached under their own combine_imputed_* columns rather than replacing the reported values.

Usage

```
add_combine_imputed(x, cols = NULL, verbose = TRUE)
```

Arguments

x	A data.frame returned by taxify().
cols	Which columns to attach: NULL (default) the curated set, "all" every column the source carries, or a character vector of names. See enrichment_cols.
verbose	Logical. Default TRUE.

Details

Source: COMBINE (Soria et al. 2021, Ecology, CC0), imputed table (trait_data_imputed.csv). Same species set and column meanings as add_combine_reported(); the difference is coverage. Because imputation only fills gaps, applying both functions and comparing combine_* with combine_imputed_* shows which values are measured versus estimated.

Value

The same data.frame with additional columns, mirroring add_combine_reported() but named combine_imputed_* (e.g. combine_imputed_adult_mass_g, combine_imputed_gestation_length_d). Traits COMBINE does not impute (for example biogeographical_realm and habitat_breadth_n) carry the reported coverage; the life-history traits (gestation, weaning, longevity, litter size, generation length, ...) are near-complete.

References

Soria CD et al. (2021) COMBINE: a coalesced mammal database of intrinsic and extrinsic traits. Ecology 102:e03344. doi:10.1002/ecy.3344

See Also

add_combine_reported() for the measured values; add_combine() for the two coalesced with per-trait provenance.

Examples

```
## Not run:
# Compare reported vs imputed gestation length for a mammal:
taxify("Vulpes vulpes", backbone = "gbif") |>
  add_combine_reported() |>
  add_combine_imputed()

## End(Not run)
```

`add_combine_reported` *Add mammal traits from COMBINE (reported values)*

Description

Joins the COMBINE mammal trait database to a `taxify()` result by looking up `accepted_name`, attaching the **reported** (directly measured or literature-compiled) trait values. COMBINE ships two parallel tables over the same ~6.2k mammal species: the reported table used here, and a phylogenetically imputed table with the missing cells filled in by a model (`add_combine_imputed()`). They are offered as two separate sources so the distinction between a measured value and a model estimate stays explicit.

Usage

```
add_combine_reported(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: COMBINE (Soria et al. 2021, Ecology, CC0), reported table. Coverage: ~6.2k mammal species, keyed on the IUCN 2020 binomial. The reported table leaves a cell missing when no source measured that trait for a species; use `add_combine_imputed()` for the higher-coverage phylogenetically imputed values.

Value

The same data.frame with additional columns (reported values):

combine_adult_mass_g Adult body mass (g).

combine_adult_body_length_mm Adult head-body length (mm).

combine_litter_size_n Litter size (count).

combine_litters_per_year_n Litters per year (count).

combine_max_longevity_d Maximum longevity (days).

combine_gestation_length_d Gestation length (days).

combine_weaning_age_d Weaning age (days).

combine_generation_length_d Generation length (days).

combine_dispersal_km Natal dispersal distance (km).

combine_habitat_breadth_n Number of IUCN habitats (count).

combine_diet_breadth_n Number of diet categories (count).

combine_trophic_level Trophic level (1 herbivore, 2 omnivore, 3 carnivore).

combine_activity_cycle Activity cycle (1 nocturnal, 2 cathemeral, 3 diurnal).

combine_foraging_stratum Foraging stratum (G/Ar/A/S/M).

combine_biogeographical_realm Biogeographical realm(s).

References

Soria CD et al. (2021) COMBINE: a coalesced mammal database of intrinsic and extrinsic traits. Ecology 102:e03344. doi:10.1002/ecy.3344

See Also

`add_combine()` for reported values with imputed gaps filled and per-trait provenance; `add_combine_imputed()` for the imputed values; `add_pantheria()`, `add_anage()` for other mammal trait sources.

Examples

```
## Not run:
taxify("Vulpes vulpes", backbone = "gbif") |>
  add_combine_reported()

## End(Not run)
```

---

add_common_names *Add common (vernacular) names*

Description

Joins vernacular names to a `taxify()` result by looking up `accepted_name`, filtered by language.

Usage

```
add_common_names(x, lang = "en", cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`lang`	Character. ISO 639-1 language code (e.g., `"en"`, `"de"`, `"fr"`), or `NA` to return names without a language tag (NCBI/OTT sources). Default `"en"`. List the languages present with `enrichment_groups("common_names")`.
`cols`	Which columns to attach. `NULL` (the default) attaches the curated set; a character vector of column names attaches just those, and `"all"` attaches every column the source carries (see `enrichment_cols()`).
`verbose`	Logical. Default `TRUE`.

Details

Common names are merged from three sources:

- GBIF backbone vernacular names (CC0), multi-language via ISO 639-1 codes.
- NCBI Taxonomy common names (public domain), no language tag (`lang = NA`).
- Open Tree of Life common names (CC0), no language tag (`lang = NA`).

When multiple common names exist for a species in the requested language, the first (most commonly used) is returned.

Value

The same data.frame with an additional column:

common_name The vernacular name in the requested language, or `NA` if none is available.

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify("Quercus robur") |>
  add_common_names()

taxify("Quercus robur") |>
  add_common_names(lang = "de")

options(old)
```

---

add_copepod_traits *Add copepod traits (Brun et al. 2017)*

Description

Joins marine-copepod traits from the trait database of Brun et al. (2017) to a `taxify()` result by looking up `accepted_name`. Records are reduced to per-species values (numeric by median, categorical by mode).

Usage

```
add_copepod_traits(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: A trait database for marine copepods (Brun et al. 2017), PANGAEA, CC BY 3.0.

Value

The same data.frame with additional columns. The default set:

cop_body_length_mm Body length (mm).

cop_egg_diameter_um Egg outer diameter (micrometres).

cop_clutch_size Clutch size.

cop_feeding_mode Feeding mode.

cop_spawning_strategy Spawning strategy.

`cols = "all"` also attaches feeder type, myelination, resting-egg presence, and the per-species min/max/n spread of the numeric values.

References

Brun P, Payne MR, Kiorboe T (2017) A trait database for marine copepods. Earth System Science Data 9:99-113. doi:10.5194/essd9992017

Examples

```
## Not run:
# Downloads the enrichment on first use.
taxify("Calanus finmarchicus", backbone = "gbif") |>
  add_copepod_traits()

## End(Not run)
```

`add_coral_traits` *Add scleractinian coral traits (Coral Trait Database)*

Description

Joins species-level coral functional traits to a `taxify()` result by looking up `accepted_name`. Values are aggregated from the long-format Coral Trait Database (numeric traits by median, categorical traits by mode).

Usage

```
add_coral_traits(x, cols = NULL, verbose = TRUE)
```

Arguments

x	A data.frame returned by taxify().
cols	Which columns to attach: NULL (default) the curated set, "all" every column the source carries, or a character vector of names. See enrichment_cols.
verbose	Logical. Default TRUE.

Details

Source: Coral Trait Database (Madin et al. 2016, Scientific Data, CC BY 4.0). Coverage: ~1.5k coral species.

Value

The same data.frame with additional columns:

coral_symbiotic_state Zooxanthellate / azooxanthellate.

coral_growth_form Typical growth form (massive/branching/...).

coral_coloniality Colonial / solitary.

coral_substrate_attachment Attached / unattached.

coral_sexual_system Hermaphrodite / gonochore.

coral_larval_development_mode Spawner / brooder.

coral_symbiont_clade Symbiodinium clade.

coral_corallite_width_max_mm Maximum corallite width (mm).

coral_colony_max_diameter_cm Maximum colony diameter (cm).

coral_growth_rate_mm_yr Linear extension rate (mm/year).

coral_depth_lower_m Lower depth limit (m).

coral_depth_upper_m Upper depth limit (m).

coral_skeletal_density_g_cm3 Skeletal density (g/cm3).

References

Madin JS et al. (2016) The Coral Trait Database, a curated database of trait information for coral species from the global oceans. Scientific Data 3:160017. doi:10.1038/sdata.2016.17

Examples

```
## Not run:
taxify("Acropora millepora", backbone = "gbif") |>
  add_coral_traits()

## End(Not run)
```

add_data *Add custom data by taxonomic matching*

Description

Joins an external data source (CSV file or data.frame) to a `taxify()` result. Species names in the external data are matched through the same backbone(s) used in the original `taxify()` call, and the join is performed on the accepted taxon — so synonyms in either dataset resolve to the same key. The key is the `accepted_id` together with the backbone that issued it, falling back to `accepted_name` where the two sides were matched by different backbones: backbone IDs are bare integers in most backbones and mean nothing outside the one they came from.

Usage

```
add_data(
  x,
  data,
  species_col = NULL,
  table = NULL,
  sheet = NULL,
  start_row = NULL,
  cols = NULL,
  group_col = NULL,
  groups = "all",
  fuzzy = TRUE,
  fuzzy_threshold = 0.2,
  verbose = TRUE
)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`data`	One of: • A **data.frame** already in R. • A **file path** to a `.csv`, `.csv.gz`, `.tsv`, `.tsv.gz`, `.xlsx`, `.sqlite`/`.db`, or `.vtr` file (read via vectra).
`species_col`	Character. Name of the column in `data` that contains species names. If `NULL` (default), auto-detected by matching `head(10)` of each character column against the backbone.
`table`	Character. Required when `data` is a SQLite file — the table name to read.
`sheet`	Integer or character. Sheet to read when `data` is an `.xlsx` file. Default `NULL` (auto-detect the sheet containing species names). Set explicitly to skip auto-detection.
`start_row`	Integer. Row where column headers begin in an `.xlsx` file. Default `NULL` (auto-detect by scanning the first 20 rows for a header row that produces species name matches). Set explicitly when the layout is known.

cols	Character vector of column names from data to join. If NULL (default), all columns except species_col are joined.
group_col	Character or NULL. Column in data that defines groups (e.g., country codes, regions). When set, the output is pivoted to wide format with one column per group (e.g., trait_AT, trait_DE), just like the built-in grouped enrichments. Use taxify_long() to reshape back to long format. Default NULL (flat join, one row per species).
groups	Character vector or "all". Which groups to include when group_col is set. Default "all".
fuzzy	Logical. Enable fuzzy matching for names in data. Default TRUE.
fuzzy_threshold	Numeric. Maximum allowed distance for fuzzy matches. Default 0.2.
verbose	Logical. Default TRUE.

Details

The workflow:

1. Read data (CSV or data.frame).
2. Identify the species column (explicit or auto-detected).
3. Match species names through the same backbone(s) as the original taxify() call, obtaining accepted_id for each row.
4. Check for conflicting duplicates: if multiple rows in data resolve to the same accepted name with different values, an error is raised (with group_col set, within each group). Exact duplicates produce a warning and are deduplicated.
5. Left-join on the accepted taxon.

Grouped data:

When data has multiple rows per species (e.g., one row per species per country), set group_col to produce wide output with suffixed columns. This is the same format as the built-in grouped enrichments.

Auto-detection:

When species_col is not specified, add_data() takes the first 10 rows of each character column and runs them through taxify(). The column with the highest match rate is selected. If no column achieves at least 50% matches, an error is raised asking the user to specify species_col explicitly.

Value

The input data.frame with additional columns from data, joined via the backbone-resolved accepted taxon. Columns from data that collide with existing columns in x are prefixed with "data_".

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

result <- taxify(c("Quercus robur", "Pinus sylvestris"))
traits <- data.frame(species = c("Quercus robur", "Pinus sylvestris"),
                     height = c(30, 25))
result |> add_data(traits, species_col = "species")

options(old)
```

`add_diaz_traits` *Add seed mass and plant height (Diaz et al. 2022)*

Description

Joins species-level mean seed mass and plant height from Diaz et al. (2022) to a `taxify()` result by looking up `accepted_name`.

Usage

```
add_diaz_traits(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Diaz et al. 2022, TRY File Archive (CC BY 3.0). Coverage: ~46k plant species. Plants only.

Value

The same data.frame with additional columns:

seed_mass_mg Seed mass in milligrams (species-level mean).

plant_height_m Plant height in metres (species-level mean).

References

Diaz S, Kattge J, Cornelissen JHC, Wright IJ, Lavorel S, Dray S, Reu B, Kleyer M, Wirth C, Prentice IC, et al. (2022) The global spectrum of plant form and function: enhanced species-level trait dataset. Scientific Data 9:755. doi:10.1038/s41597022017749

Diaz S, Kattge J, Cornelissen JHC, Wright IJ, Lavorel S, Dray S, Reu B, Kleyer M, Wirth C, Prentice IC, et al. (2016) The global spectrum of plant form and function. Nature 529:167-171. doi:10.1038/nature16489

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify("Quercus robur") |>
  add_diaz_traits()

options(old)
```

`add_disperse` *Add aquatic-invertebrate dispersal traits (DISPERSE)*

Description

Joins genus-level dispersal-related traits for European aquatic macroinvertebrates to a `taxify()` result by `genus`. Each fuzzy-coded trait is reduced to its dominant modality (with the database's own labels).

Usage

```
add_disperse(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: DISPERSE (Sarremejane et al. 2020, Scientific Data, CC-BY 4.0). Joins on genus because the database is genus-resolved.

Value

The same data.frame with categorical `disperse_body_size_cm`, `disperse_life_cycle`, `disperse_repro_cycles`, `disperse_dispersal`, `disperse_adult_lifespan`, `disperse_female_wing_mm`, `disperse_wing_type`, `disperse_fecundity`, `disperse_drift`, and the numeric bin-midpoint columns `disperse_body_size_cm_mid`, `disperse_female_wing_mm_mid`, `disperse_fecundity_mid` (all joined on genus).

References

Sarremejane R et al. (2020) DISPERSE, a trait database to assess the dispersal potential of European aquatic macroinvertebrates. Scientific Data 7:386. doi:10.6084/m9.figshare.c.5000633

Examples

```
## Not run:
taxify("Baetis rhodani", backbone = "gbif") |>
  add_disperse()

## End(Not run)
```

`add_eberswalde` *Add Eberswalde long-term carabid monitoring traits and trends*

Description

Joins ground-beetle traits and 24-year local population trends to a `taxify()` result by accepted name.

Usage

```
add_eberswalde(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) all of them, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Weiss, von Wehrden & Linde (2024), 27 species pitfall-trapped on 13 forest plots near Eberswalde between 1999 and 2022.

`body_length_mm`, `wing_morph` and `range_centre_lat` are carabids.org (Homburg et al. 2014) verbatim, which the deposit's README states and the data confirm: over the 19 species shared with `add_chowdhury()` every size is exactly equal and every wing class agrees. They are surfaced

here because each add_<source>() function shows what its source carries, and they are kept out of add_trait(), where they would double-count one lineage.

What belongs to this deposit alone is the monitoring result: abundance_trend over the 24 years, drought_effect against the 72-month SPEI index, and a feeding_guild refined by the authors' field observations to name the prey. humidity_pref is Sustek's (2004) 1-8 scale, 1 dry to 8 humid.

Value

The same data.frame with eberswalde_ columns: numeric body_length_mm, humidity_pref, range_centre_lat, abundance_total; categorical wing_morph, feeding_guild, abundance_trend, drought_effect.

References

Weiss F, von Wehrden H, Linde A (2024) Eberswalde Carabid Monitoring 1999-2022 - Full Data. PubData Leuphana. doi:10.48548/pubdata46

Weiss F, von Wehrden H, Linde A (2024) Long-term drought triggers severe declines in carabid beetles in a temperate forest. Ecography 2024(4):e07020. doi:10.1111/ecog.07020

Examples

```
## Not run:
taxify(c("Carabus coriaceus", "Nebria brevicollis")) |>
  add_eberswalde()

## End(Not run)
```

---

add_ecoflora *Add British plant traits from Ecoflora*

Description

Joins traits from the Ecological Flora of the British Isles (Fitter & Peat 1994) to a taxify() result by looking up accepted_name. Ecoflora covers the vascular flora of the British Isles, providing canopy height, leaf traits, life form, flowering phenology, pollination and reproduction, seed weight, and British-calibrated Ellenberg indicator values. Every column carries a _uk suffix to mark the British-flora calibration and to avoid collisions when chained with other plant-trait enrichments (e.g. add_baseflor() for France, add_floraweb() for Germany).

Usage

```
add_ecoflora(x, cols = NULL, verbose = TRUE)
```

Arguments

x	A data.frame returned by taxify().
cols	Which columns to attach: NULL (default) the curated set, "all" every column the source carries, or a character vector of names. See enrichment_cols.
verbose	Logical. Default TRUE.

Details

Source: Ecoflora (Ecological Flora of the British Isles). Ecoflora has no bulk download or API; the bundled dataset was collected one species at a time and is redistributed under the source licence (CC BY-NC-SA 4.0). The .vtr is downloaded from the taxify release on first use and cached.

For French-flora traits see add_baseflor(); for German-flora traits see add_floraweb(); for European-calibration indicator values see add_eive().

Value

The same data.frame with additional _uk columns:

height_max_mm_uk, height_min_mm_uk Typical maximum and minimum plant height (mm; Ecoflora records cm).

leaf_area_uk Leaf area class (cm^2).

leaf_longevity_uk Leaf longevity (e.g. evergreen, deciduous).

root_system_uk Root system type.

photosynthetic_pathway_uk Photosynthetic pathway (C3/C4/CAM).

life_form_uk Raunkiaer life form.

reproduction_uk Reproduction method.

flower_begin_month_uk, flower_end_month_uk Flowering months (1-12).

pollination_vector_uk Pollen vector(s).

seed_weight_mg_uk Seed weight (mg).

propagule_uk Propagule / dispersule type.

ell_light_uk, ell_moisture_uk, ell_reaction_uk, ell_nitrogen_uk, ell_salt_uk Ellenberg indicator values calibrated for the British flora (light, moisture, reaction, nitrogen, salt).

References

Fitter AH, Peat HJ (1994) The Ecological Flora Database. Journal of Ecology 82:415-425.

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify("Bellis perennis") |>
  add_ecoflora()

options(old)
```

`add_ecomorphosis` *Add Collembola ecomorphosis (Bonfanti et al. 2022)*

Description

Joins the ecomorphosis record of a springtail species to a `taxify()` result by looking up `accepted_name`. Ecomorphosis is a seasonal, reversible change of form some Collembola undergo; this marks the species known to display it.

Usage

```
add_ecomorphosis(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) all, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: extended species list of Collembola known to display ecomorphosis, Bonfanti (2022), Zenodo, CC BY 4.0. Coverage: 43 species, each carrying the literature record for its ecomorphic form. It is a presence list: a species not listed is not evidence of absence.

Value

The same data.frame with additional columns:

ecomorphosis Whether the species is known to display ecomorphosis.

ecomorphosis_area The morphological area affected.

ecomorphosis_reference The literature record establishing it.

References

Bonfanti J, Krogh PH, Hedde M, Cortet J (2022) Ecomorphosis in European Collembola: a review in the context of trait-based ecology. Applied Soil Ecology. doi:10.1016/j.apsoil.2022.104692

Examples

```
## Not run:
taxify("Hypogastrura vesiculosa", backbone = "gbif") |>
  add_ecomorphosis()

## End(Not run)
```

`add_edwards_phyto` *Add phytoplankton nutrient-uptake traits (Edwards et al.)*

Description

Joins species-level phytoplankton nutrient physiology (Droop/Monod uptake and growth parameters for ammonium, nitrate and phosphorus, plus cell size and carbon content) to a `taxify()` result by looking up `accepted_name`.

Usage

```
add_edwards_phyto(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Edwards et al. (2015, Ecology, CC BY 4.0), a compilation of phytoplankton nutrient-utilization traits for ~130 species. Single-source physiological data with no cross-source analogue, so it is attached by this function and not registered with `add_trait()`.

Value

The same data.frame with additional columns. The curated set:

edwards_taxon_group Coarse phytoplankton group (diatom, green, ...).

edwards_habitat_system Habitat (marine/freshwater).

edwards_cell_volume Cell volume (micron^3).

edwards_carbon_per_cell Carbon content per cell (pg C).

edwards_mu_inf_nit Maximum growth rate on nitrate (per day).

edwards_k_nit Half-saturation constant for growth on nitrate.

edwards_qmin_nit Minimum cell nitrogen quota.

edwards_mu_inf_p Maximum growth rate on phosphorus (per day).

edwards_k_p Half-saturation constant for growth on phosphorus.

edwards_qmin_p Minimum cell phosphorus quota.

With `cols = "all"` the full set of ammonium/nitrate/phosphorus uptake and quota parameters (`vmax_amm`, `mu_nit`, `vmax_p`, `qmax_p`, ...) is attached under their source names. All uptake/quota traits are joined on `accepted_name`.

References

Edwards KF, Thomas MK, Klausmeier CA, Litchman E (2015) Phytoplankton growth and the interaction of light and temperature: A synthesis at the species and community level. Ecology 96(9):2554-2564. doi:10.1890/142252.1

Examples

```
## Not run:
taxify("Thalassiosira pseudonana", backbone = "gbif") |>
  add_edwards_phyto()

## End(Not run)
```

`add_eive` *Add EIVE ecological indicator values*

Description

Joins EIVE 1.0 (Dengler et al. 2023) ecological indicator values to a `taxify()` result by looking up `accepted_name`. EIVE provides continuous indicator values for European vascular plants, superseding the original ordinal Ellenberg values.

Usage

```
add_eive(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: EIVE 1.0 (Dengler et al. 2023, Zenodo, CC BY 4.0). Coverage: ~14.5k European vascular plant species.

Value

The same data.frame with additional columns:

eive_light Light indicator value (continuous).

eive_temperature Temperature indicator value (continuous).

eive_moisture Moisture indicator value (continuous).

eive_reaction Soil reaction (pH) indicator value (continuous).

eive_nutrients Nutrient indicator value (continuous).

References

Dengler J et al. (2023) EIVE 1.0: a standardized set of Ecological Indicator Values for Europe. Vegetation Classification and Survey 4:7-29. doi:10.3897/VCS.98324

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify("Arrhenatherum elatius") |>
  add_eive()

options(old)
```

`add_ellers_collembola` *Add Collembola traits (Ellers et al. 2018)*

Description

Joins European springtail functional traits to a `taxify()` result by looking up `accepted_name`: vertical stratification, body size, reproduction mode and climatic preferences.

Usage

```
add_ellers_collembola(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) all, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Collembola trait table from Ellers et al. (2018), Dryad, CC0. Coverage: 278 European Collembola species; taxonomy follows the Checklist of the Collembola of the World. The openly licensed analogue of the BETSI multi-trait matrices; its body size agrees with the BETSI body-length floor at Pearson r = 0.96 over 262 shared species. The body size is also available through `add_trait()("body_length")`.

Value

The same data.frame with additional columns:

ellers_vertical_distribution Vertical stratification (soil depth class).

ellers_body_size_mm Body size (mm).

ellers_reproduction Reproduction mode (sexual or asexual).

ellers_moisture_pref Moisture preference.

ellers_temperature_pref Temperature preference.

ellers_thermal_niche_breadth Thermal niche breadth.

References

Ellers J, Berg MP, Dias ATC, Fontana S, Ooms A, Moretti M (2018) Diversity in form and function: vertical distribution of soil fauna mediates multidimensional trait variation. Journal of Animal Ecology 87:933-944. doi:10.1111/13652656.12838

Examples

```
## Not run:
taxify("Isotoma viridis", backbone = "gbif") |>
  add_ellers_collembola()

## End(Not run)
```

`add_elton_traits` *Add diet, foraging, and body mass (EltonTraits 1.0)*

Description

Joins EltonTraits 1.0 diet composition, foraging strata, body mass, and activity data to a `taxify()` result by looking up `accepted_name`.

Usage

```
add_elton_traits(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: EltonTraits 1.0 (Wilman et al. 2014, Figshare, CC0). Coverage: ~15.4k species. Birds and mammals only.

Value

The same data.frame with additional columns:

diet_inv Percentage of diet: invertebrates.

diet_vend Percentage of diet: endothermic vertebrates.

diet_vect Percentage of diet: ectothermic vertebrates.

diet_vfish Percentage of diet: fish.

diet_vunk Percentage of diet: unknown vertebrates.

diet_scav Percentage of diet: scavenging.

diet_fruit Percentage of diet: fruit.

diet_nect Percentage of diet: nectar.

diet_seed Percentage of diet: seeds and nuts.

diet_plantother Percentage of diet: other plant material.

foraging_water Percentage of foraging: below water surface.

foraging_ground Percentage of foraging: on ground.

foraging_understory Percentage of foraging: in understory.

foraging_midhigh Percentage of foraging: in mid to high strata.

foraging_canopy Percentage of foraging: in canopy.

foraging_aerial Percentage of foraging: aerial.

elton_body_mass_g Body mass in grams.

nocturnal Nocturnal activity (0 = diurnal, 1 = nocturnal).

diet_guild Dominant diet guild derived from the ten diet-fraction columns (fractions summed within guild, the guild reaching 50 percent wins, otherwise omnivore): carnivore, herbivore, omnivore, invertivore, frugivore, granivore, nectarivore, scavenger. Also reachable across sources via `add_trait(x, "diet_guild")`.

References

Wilman H et al. (2014) EltonTraits 1.0: Species-level foraging attributes of the world's birds and mammals. Ecology 95:2027.

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify("Parus major", backbone = "gbif") |>
  add_elton_traits()

options(old)
```

`add_epa_freshwater` *Add freshwater invertebrate traits (US EPA)*

Description

Joins primary functional traits of freshwater macroinvertebrates from the U.S. EPA Freshwater Biological Traits Database to a `taxify()` result. Records are reduced to per-taxon modes. The database records each trait at the finest available resolution, so the join matches `accepted_name` first and then fills any trait still missing from the taxon's genus-level row; a species-level value is never overwritten.

Usage

```
add_epa_freshwater(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: U.S. EPA Freshwater Biological Traits Database (2012), a public-domain U.S. Government work, compiled primarily from Vieira et al. (2006).

Value

The same data.frame with additional columns. The default set:

epa_feeding_mode Functional feeding group.

epa_habit Primary habit (e.g. clinger, burrower, swimmer).

epa_voltinism Number of generations per year.

epa_thermal_preference Thermal preference.

epa_body_size_class Maximum body-size class.

`cols = "all"` also attaches body shape, rheophily, oviposition behaviour, and diapause.

References

U.S. Environmental Protection Agency (2012) Freshwater Biological Traits Database. Compiled from Vieira NKM et al. (2006) A database of lotic invertebrate traits for North America, USGS Data Series 187.

Examples

```
## Not run:
# Downloads the enrichment on first use.
taxify("Baetis", backbone = "gbif") |>
  add_epa_freshwater()

## End(Not run)
```

`add_eupolltrait` *Add European pollinator traits (EuPollTrait)*

Description

Joins European bee and hoverfly morphological, biogeographic and ecological traits to a `taxify()` result by `accepted_name`.

Usage

```
add_eupolltrait(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: EuPollTrait (Milicic et al. 2025, Zenodo, CC-BY 4.0).

Value

The same data.frame with `eupolltrait_` columns: numeric `itd_mm`, `tongue_length_mm`, `species_temperature_index`, `species_continentality_index`, `area_of_occupancy`, `extent_of_occurrence`; categorical `sociality`, `nest`, `larval_nutrition`, `body_length_category`.

References

Milicic M et al. (2025) EuPollTrait: a trait database for European bees and hoverflies. Zenodo. doi:10.5281/zenodo.18032357

Examples

```
## Not run:
taxify("Bombus terrestris", backbone = "gbif") |>
  add_eupolltrait()

## End(Not run)
```

`add_eurobat` *Add European bat traits (EuroBaTrait)*

Description

Joins species-level traits of European bats (morphology, life history, diet, foraging habitat, roost type) to a `taxify()` result by looking up `accepted_name`.

Usage

```
add_eurobat(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Froidevaux et al. (2023, Scientific Data, CC BY 4.0), EuroBaTrait 1.0. Thematic measurement-or-fact tables are reduced to species-level values (numeric by median, categorical by mode).

Value

The same data.frame with additional columns. The curated set:

eurobat_forearm_length_mm Forearm length (mm).

eurobat_body_mass_g Body mass (g).

eurobat_max_longevity_yr Maximum recorded longevity (years).

eurobat_litter_size Litter size.

eurobat_diet_type Diet type (insectivorous, frugivorous, ...).

eurobat_first_main_prey First main prey item.

With `cols = "all"` the full trait set (digit lengths, wing indices, habitat affinity scores, critical feeding areas, roost dependence, phenology, ...) is attached under their source names.

References

Froidevaux JSP et al. (2023) EuroBaTrait 1.0: a species-level trait dataset of bats in Europe and beyond. Scientific Data. figshare doi:10.6084/m9.figshare.21777161

Examples

```
## Not run:
taxify("Myotis myotis", backbone = "gbif") |>
  add_eurobat()

## End(Not run)
```

`add_euromed_distribution` *Add Euro+Med PlantBase distribution status*

Description

Joins the per-area status Euro+Med PlantBase states for a taxon (native, naturalised, introduced, casual, ...) to a `taxify()` result, for the areas or countries asked for. The names and the ids of the `euromed` backbone are the same taxa the status is recorded on, so a result matched against any backbone joins through its accepted name.

Usage

```
add_euromed_distribution(x, region, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`region`	Character. ISO 3166-1 alpha-2 country code(s), Euro+Med area code(s), or `"all"`. A code that is an ISO code of a Euro+Med area is read as the country; anything else must be an area code, as listed by `euromed_areas()`. • Single region: adds `euromed_status` (no suffix). • Several regions: adds `euromed_status_<code>` per region. • `"all"`: one column per area.
`cols`	Which columns to attach. `NULL` (the default) attaches `euromed_status`; a character vector attaches just those, and `"all"` attaches every column the source carries (see `enrichment_cols()`), among them `euromed_status_detail`, `euromed_status_source`, `area_name`, `iso2` and the taxon's Euro+Med UUID, `taxon_id`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Euro+Med PlantBase (CC BY-SA), European and Mediterranean vascular plants, bryophytes and some fungi and algae. A report Euro+Med marks as made in error is not carried; a former presence is `"extinct"`. Where the records of one taxon and area disagree, the stronger claim wins in the order given above (a taxon is native wherever a record says so), and the references are those of the records stating it. The country of an area comes from its own name, so a combined area has none and its countries are read from the level-2 areas.

Value

The same data.frame with `euromed_status`, one of `"native"`, `"naturalised"`, `"introduced"`, `"casual"`, `"cultivated"`, `"doubtful"` or `"extinct"`, and `NA` where Euro+Med states nothing for the region. `euromed_status_detail` carries the term as Euro+Med words it (`"endemic"`, `"introduced: uncertain degree of naturalisation"`). `euromed_status_source` holds the ids of the citing references; resolve them with `cite(x$euromed_status_source, source = "euromed_distribution")`.

See Also

`euromed_areas()`, `add_wcvp()`

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify("Solidago canadensis") |>
  add_euromed_distribution(region = "DE")

options(old)
```

`add_faprotax` *Add prokaryote metabolic and ecological functions (FAPROTAX)*

Description

Joins the function groups a prokaryotic taxon is known to perform (methanogenesis, denitrification, nitrogen fixation, chitinolysis, human gut association and 87 others) to a `taxify()` result.

Usage

```
add_faprotax(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) both, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Louca et al. (2016), 92 function groups over 4470 taxa compiled from IJSEM and Bergey's Manual. Redistributed under FAPROTAX's own BSD-style terms; see `cite` for the notice that must travel with it.

FAPROTAX annotates a taxon at whatever rank the evidence supports, so its entries are a mix of species and genera. The join follows: a species-level entry matches the accepted name, and a taxon with no entry of its own inherits its genus's. Group memberships are the source's; taxifydb reshapes the grouped list into one row per taxon and reduces each entry to the species or genus it names, which the licence requires be stated as a modification.

Functions stay a set rather than one label because a prokaryote genuinely performs several: *Escherichia coli* carries 17 of them.

Value

The same data.frame with `faprotax_functions` (a `|`-delimited set) and `faprotax_n_functions`.

References

Louca S, Parfrey LW, Doebeli M (2016) Decoupling function and taxonomy in the global ocean microbiome. Science 353:1272-1277. doi:10.1126/science.aaf4507

Examples

```
## Not run:
taxify(c("Escherichia coli", "Nitrosomonas europaea")) |>
  add_faprotax()

## End(Not run)
```

---

add_finand *Add Helsinki urban-forest carabid traits (Finand & Kotze)*

Description

Joins ground-beetle traits measured from Helsinki specimens to a `taxify()` result by accepted name.

Usage

```
add_finand(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) all of them, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Finand & Kotze (2025), 34 species from 25 remnant urban forests in Helsinki.

Small, and carried for a reason no larger source can supply. Every other reachable carabid trait table traces its body sizes and wing classes back to carabids.org (Homburg et al. 2014), so agreement between them is not corroboration. These body lengths were measured from the beetles the authors caught, which makes them the one independent check: they run at a median ratio of 0.9946 against the NW European arthropod compilation over 28 shared species, with only 7.1 percent exactly equal.

`body_length_mm` is a whole-body length despite the source column being terse about it; elytra length would sit near 0.6 of these values.

`wing_morph` records the morph of the beetles caught in Helsinki, so it can name a definite morph where a species-level source calls the species dimorphic. Compare `add_chowdhury()`, which states the species' capacity.

Value

The same data.frame with `finand_` columns: numeric `body_length_mm`; categorical `wing_morph`, `feeding_type`, `habitat_pref`, `moisture_pref`.

References

Finand B, Kotze DJ (2025) Habitat specialisation and dispersal capacity drive rapid carabid beetle responses to urban forest fragmentation. Zenodo. doi:10.5281/zenodo.17184995

Examples

```
## Not run:
taxify(c("Carabus glabratus", "Cychrus caraboides")) |>
  add_finand()

## End(Not run)
```

---

add_fishbase *Add fish traits (FishBase)*

Description

Joins FishBase morphological and ecological traits to a `taxify()` result by looking up `accepted_name`.

Usage

```
add_fishbase(x, cols = NULL, verbose = TRUE)
```

Arguments

x	A data.frame returned by taxify().
cols	Which columns to attach: NULL (default) the curated set, "all" every column the source carries, or a character vector of names. See enrichment_cols.
verbose	Logical. Default TRUE.

Details

Source: FishBase via rfishbase (Froese & Pauly, CC BY-NC 4.0). Coverage: ~35k fish species. Fishes only.

The fb_lw_* columns give one representative length-weight fit per species from the POPLW table, so a length can be converted to a mass where fb_body_mass_g (a record maximum) does not fit the analysis. FishBase's Bayesian congeneric estimates are not included; fb_lw_* are measured fits.

The build-from-source fallback requires the **rfishbase** package (available on CRAN). Pre-built .vtr files do not require rfishbase.

Value

The same data.frame with additional columns:

fb_body_length_cm Maximum body length in centimetres.

fb_body_mass_g Maximum published weight in grams (FishBase SPECIES.Weight). This is a record maximum, not a typical or adult-mean mass.

fb_trophic_level Trophic level.

fb_depth_min_m Minimum depth in metres.

fb_depth_max_m Maximum depth in metres.

fb_vulnerability Vulnerability index (0–100).

fb_habitat Habitat type (e.g. demersal, pelagic).

fb_importance Commercial importance category.

fb_lw_a Coefficient a of the length-weight relationship W = a * L^b (weight in g, length in cm of type fb_lw_type), from FishBase's POPLW table.

fb_lw_b Exponent b of the length-weight relationship.

fb_lw_type Length convention the coefficients were fitted against (TL total, SL standard, FL fork, WD width, ...). Chosen to match the species' maximum-length type where recorded, so it applies to fb_body_length_cm. Applying a coefficient to a different length type is a silent error.

fb_lw_method How the fit was obtained (e.g. "type I linear regression", "single L-W pair with b=3").

fb_lw_sex Sex the fit applies to (unsexed, mixed, female, male, juvenile).

fb_lw_n Sample size the fit was based on.

fb_lw_r2 Coefficient of determination (r^2) of the fit.

References

Froese R, Pauly D (eds.) (2024) FishBase. World Wide Web electronic publication, https://www.fishbase.org.

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify("Gadus morhua", backbone = "gbif") |>
  add_fishbase()

options(old)
```

`add_fishmorph` *Add freshwater fish morphological traits (FISHMORPH)*

Description

Joins FISHMORPH morphological trait data to a `taxify()` result by looking up `accepted_name`. This function is named after its source, FISHMORPH; for the fish reference database FishBase see `add_fishbase()`.

Usage

```
add_fishmorph(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: FISHMORPH (Brosse et al. 2021, Figshare, CC BY 4.0). Coverage: ~9k freshwater fish species.

Value

The same data.frame with additional columns:

fish_max_body_length Maximum body length (cm).

fish_body_elongation Body elongation (body length / body depth).

fish_vertical_eye_position Vertical eye position (eye position / head depth).

fish_relative_eye_size Relative eye size (eye diameter / head length).

fish_oral_gape_position Oral gape position (mouth position: 0 = inferior, 0.5 = terminal, 1 = superior).

fish_relative_maxillary_length Relative maxillary length (maxillary length / head length).

fish_body_lateral_shape Body lateral shape (body depth / caudal peduncle depth).

fish_pectoral_fin_position Pectoral fin vertical position (fin insertion depth / body depth).

fish_pectoral_fin_size Pectoral fin size (fin length / body length).

fish_caudal_peduncle_throttling Caudal peduncle throttling (caudal peduncle depth / caudal fin depth).

References

Brosse S, Charpin N, Su G, Toussaint A, Herrera-R GA, Tedesco PA, Villeg ér S (2021) FISHMORPH: A global database on morphological traits of freshwater fishes. Global Ecology and Biogeography 30:2330-2336. doi:10.1111/geb.13395

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify("Salmo trutta", backbone = "gbif") |>
  add_fishmorph()

options(old)
```

---

add_fishtraits *Add United States freshwater fish traits (FishTraits)*

Description

Joins ecological and life-history traits of United States freshwater fishes from FishTraits (Frimpong & Angermeier 2009) to a `taxify()` result by looking up `accepted_name`.

Usage

```
add_fishtraits(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: FishTraits v14.3 (USGS ScienceBase), a public-domain U.S. Government work. North American freshwater fishes.

Value

The same data.frame with additional columns. The default set:

ft_common_name Common name.

ft_native Whether native to the contiguous United States.

ft_max_length_cm Maximum total length (cm).

ft_longevity_yr Maximum reported age (years).

ft_maturity_age_yr Age at maturity (years).

ft_fecundity_max Maximum fecundity.

ft_repro_guild Reproductive guild.

ft_min_temp_c Lower temperature tolerance (deg C).

ft_max_temp_c Upper temperature tolerance (deg C).

ft_extinct Whether recorded as extinct.

`cols = "all"` also attaches the ten diet-category flags, salinity tolerance, flow preferences, migratory strategy, listing status, and the ITIS TSN.

References

Frimpong EA, Angermeier PL (2009) FishTraits: a database of ecological and life-history traits of freshwater fishes of the United States. Fisheries 34:487-495.

Examples

```
## Not run:
# Downloads the enrichment on first use.
taxify("Micropterus salmoides", backbone = "gbif") |>
  add_fishtraits()

## End(Not run)
```

`add_floraweb` *Add German plant traits from FloraWeb*

Description

Joins traits from FloraWeb (Bundesamt fuer Naturschutz) to a `taxify()` result by looking up `accepted_name`. FloraWeb is the live national portal carrying the BiolFlor trait data (Klotz, Kuehn & Durka 2002) together with Rothmaler morphology and Ellenberg indicator values. This enrichment covers the full per-species trait profile scraped from the four FloraWeb trait pages: morphology, reproductive biology, the nine Ellenberg indicator values, ploidy and chromosome number, and chorological distribution. Every column carries a `_de` suffix to mark the German-flora calibration and to avoid collisions when chained with other plant-trait enrichments (e.g. `add_ecoflora()` for Britain, `add_baseflor()` for France).

Usage

```
add_floraweb(x, cols = NULL, verbose = TRUE)
```

Arguments

x	A data.frame returned by taxify().
cols	Which FloraWeb columns to attach. NULL (the default) attaches all of them; a character vector of _de column names attaches just those (see enrichment_cols(), or the columns listed below). Pass "all" for the full set explicitly.
verbose	Logical. Default TRUE.

Details

Source: FloraWeb (https://www.floraweb.de/), Bundesamt fuer Naturschutz, Bonn. FloraWeb has no bulk export or API; the bundled dataset was scraped per species (accessed 2026-06-24) and that access date is the dataset version. The trait data largely derive from BiolFlor, which per the BioFresh metadata statement is publicly available and may be used without restrictions provided it is acknowledged and cited correctly. The .vtr is downloaded from the taxify release on first use and cached.

For British-flora traits see add_ecoflora(); for French-flora traits see add_baseflor(); for European-calibration indicator values see add_eive().

Value

The same data.frame with German trait columns (all suffixed _de), grouped as:

Morphology height_de, life_form_de, leaf_shape_de, leaf_anatomy_de, leaf_persistence_de, storage_organs_de, flowering_months_de, flowering_months_biolflor_de, flowering_phase_de, phenological_season_de, description_de.

Reproductive biology pollination_vector_de, pollinator_de, pollinator_reward_de, flower_type_de, flower_class_de, dispersal_type_de, diaspore_type_de, germinule_type_de, reproduction_type_de, vegetative_spread_de, fertilization_type_de, apomixis_de, dicliny_de, dichogamy_de, self_incompatibility_de, si_mechanism_de, ploidy_de, chromosome_number_de, chromosome_freq_de, chromosomes_de.

Ecology the nine Ellenberg indicator values ell_light_de, ell_temperature_de, ell_continentality_de, ell_moisture_de, ell_moisture_variability_de, ell_reaction_de, ell_nitrogen_de, ell_salt_de, heavy_metal_resistance_de, plus strategy_type_de (Grime CSR), habitat_site_de, formation_de, plant_community_de, biotope_type_de, forest_binding_de, hemeroby_de, urbanity_de.

Distribution floristic_zones_de, areal_formula_de, areal_type_de, oceanity_de, range_centre_de, world_range_size_de, world_range_frequency_de, world_range_position_de, world_range_hazard_de, germany_range_share_de, germany_responsibility_de.

Categorical traits with several applicable values are joined with "; ". Trait values are German (as published by FloraWeb / BiolFlor).

References

Klotz S, Kuehn I, Durka W (2002) BIOLFLOR - Eine Datenbank zu biologisch-oekologischen Merkmalen der Gefaesspflanzen in Deutschland. Schriftenreihe fuer Vegetationskunde 38. Bundesamt fuer Naturschutz, Bonn.

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify("Bellis perennis") |>
  add_floraweb()

options(old)
```

`add_freshwater_insects_conus` *Add freshwater-insect genus traits (Freshwater Insects CONUS)*

Description

Joins genus-level ecological and life-history trait modalities of North American freshwater insects to a `taxify()` result by looking up genus, so any species in a covered genus is annotated. The modalities are the source's own abbreviation codes, kept verbatim.

Usage

```
add_freshwater_insects_conus(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Twardochleb et al. (2021, Environmental Data Initiative, CC BY 4.0), the Freshwater Insects CONUS genus trait table. Traits are genus-level categorical modalities.

Value

The same data.frame with additional columns. The curated set:

fwinsect_thermal_pref Thermal preference class.

fwinsect_feed_prim Primary functional feeding group code.

fwinsect_habit_prim Primary habit (swimmer, clinger, burrower, ...).

fwinsect_rheophily Rheophily (current preference) code.

fwinsect_voltinism Voltinism (generations per year) code.

fwinsect_max_body_size Maximum body size class.

With `cols = "all"` the remaining trait groups (emergence, dispersal, respiration, ...) are attached under their source names.

References

Twardochleb LA et al. (2021) Freshwater insect occurrences and traits for the contiguous United States, 2001-2018. Environmental Data Initiative. doi:10.6073/pasta/8238ea9bc15840844b3a023b6b6ed158

Examples

```
## Not run:
taxify("Baetis", backbone = "gbif") |>
  add_freshwater_insects_conus()

## End(Not run)
```

`add_frugivoria` *Add Neotropical frugivore traits (Frugivoria)*

Description

Joins shared bird/mammal frugivore traits to a `taxify()` result by `accepted_name`.

Usage

```
add_frugivoria(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Gerstner et al. (2023) Frugivoria (EDI, CC-BY 4.0).

Value

The same data.frame with `frugivoria_` columns: categorical `taxon_group`, `diet_category`; numeric `diet_breadth`, `body_mass_g`, `body_size_mm`, `longevity`, `generation_time`.

References

Gerstner BE et al. (2023) Frugivoria: a trait database for birds and mammals exhibiting frugivory across contiguous Neotropical moist forests. EDI (edi.1220.5).

Examples

```
## Not run:
taxify("Ramphastos toco", backbone = "gbif") |>
  add_frugivoria()

## End(Not run)
```

---

add_fungalroot *Add mycorrhizal type from FungalRoot*

Description

Joins genus-level mycorrhizal type from the FungalRoot database (Soudzilovskaia et al. 2020) to a `taxify()` result by looking up `genus`. Mycorrhizal type is phylogenetically conserved at the genus level, which is the resolution FungalRoot recommends for inference, so this enrichment joins on `genus` rather than `accepted_name`.

Usage

```
add_fungalroot(x, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`verbose`	Logical. Default `TRUE`.

Details

Source: FungalRoot, published on GBIF as a Darwin Core Archive (doi:10.15468/a7ujmj), CC BY-NC 4.0. The per-genus value is a majority consensus computed from the per-observation mycorrhiza type labels, not FungalRoot's own published per-genus assignment. Plant genera only. The `.vtr` is downloaded from the taxify release on first use and cached.

Value

The same data.frame with three additional columns:

mycorrhizal_type Genus-level majority-consensus type, one of `AM` (arbuscular), `EcM` (ecto), `ErM` (ericoid), `OM` (orchid), `NM` (non-mycorrhizal), the dual types `EcM-AM` / `ErM-EcM` / `ErM-AM`, `Other`, or `uncertain`. `NA` if the genus is not in FungalRoot.

mycorrhizal_status Coarse status derived from the type: `"mycorrhizal"`, `"non-mycorrhizal"`, or `"uncertain"`.

mycorrhizal_records Number of FungalRoot observations supporting the genus-level consensus.

References

Soudzilovskaia NA et al. (2020) FungalRoot: global online database of plant mycorrhizal associations. New Phytologist 227:955-966.

Examples

```
## Not run:
# Joins on genus, so any species in a covered genus is annotated.
taxify(c("Quercus robur", "Trifolium pratense")) |>
  add_fungalroot()

## End(Not run)
```

---

add_fungal_traits *Add fungal lifestyle and trait data (FungalTraits)*

Description

Joins FungalTraits (Polme et al. 2020) genus-level trait data to a `taxify()` result by looking up `genus`. Unlike other enrichments that join on species-level `accepted_name`, FungalTraits is a genus-level database and joins on the `genus` column already present in taxify output.

Usage

```
add_fungal_traits(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: FungalTraits (Polme et al. 2020, Fungal Diversity, CC BY 4.0). Coverage: ~10k fungal genera. Genus-level only (not species-level).

Value

The same data.frame with additional columns:

primary_lifestyle Primary ecological role (e.g., saprotroph, mycorrhizal, pathogen, endophyte, lichenized, parasite).

secondary_lifestyle Secondary ecological role, if any.

growth_form Morphological growth form (e.g., agaricoid, corticioid, polyporoid, yeast).

fruitbody_type Fruiting body morphology (e.g., gasteroid, pileate, resupinate).

decay_substrate Substrate type for saprotrophic genera (e.g., wood, litter, dung, soil).

plant_pathogenic_capacity Capacity to cause plant disease (e.g., high, medium, low, none).

animal_biotrophic_capacity Capacity for animal biotrophy.

endophytic_interaction_capability Capacity for endophytic interactions with plants.

ectomycorrhiza_exploration_type Exploration type for ectomycorrhizal genera (e.g., contact, short, medium, long).

References

Polme S et al. (2020) FungalTraits: a user-friendly traits database of fungi and fungus-like stramenopiles. Fungal Diversity 105:1-16. doi:10.1007/s13225-020-00466-2

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify("Amanita muscaria", backbone = "gbif") |>
  add_fungal_traits()

options(old)
```

`add_funguild` *Add fungal functional guild data (FUNGuild)*

Description

Joins FUNGuild trophic mode, guild, growth morphology, and confidence data to a `taxify()` result by looking up `accepted_name`. Species-level matches take priority; genus-level guild assignments are used as fallback for unmatched species.

Usage

```
add_funguild(x, cols = NULL, verbose = TRUE)
```

Arguments

x	A data.frame returned by taxify().
cols	Which columns to attach: NULL (default) the curated set, "all" every column the source carries, or a character vector of names. See enrichment_cols.
verbose	Logical. Default TRUE.

Details

Source: FUNGuild (Nguyen et al. 2016, CC BY 4.0). Coverage: ~13k taxa. Fungi only.

The enrichment first attempts species-level matching. For species without a direct match, it falls back to genus-level guild assignments from FUNGuild's genus-rank entries.

Value

The same data.frame with additional columns:

trophic_mode Trophic mode (e.g., Pathotroph, Saprotroph, Symbiotroph, or hyphenated combinations).

guild Functional guild (e.g., "Ectomycorrhizal", "Plant Pathogen", "Wood Saprotroph").

funguild_growth_form Growth morphology (e.g., "Agaricoid", "Microfungus"). Prefixed to avoid collision with FungalTraits.

confidence_ranking Confidence of the guild assignment (Possible, Probable, Highly Probable).

References

Nguyen NH et al. (2016) FUNGuild: An open annotation tool for parsing fungal community datasets by ecological guild. Fungal Ecology 20:241-248.

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify("Amanita muscaria", backbone = "gbif") |>
  add_funguild()

options(old)
```

`add_gbif_info` *Add GBIF-specific columns*

Description

Joins extra GBIF backbone columns to a `taxify()` result by looking up `taxon_id` in the GBIF backbone. Only enriches rows where `backbone == "gbif"`.

Usage

```
add_gbif_info(x)
```

Arguments

`x`	A data.frame returned by `taxify()` with `backbone == "gbif"`.

Value

The same data.frame with additional columns:

notho_type Hybrid type: `"GENERIC"`, `"SPECIFIC"`, or `"INFRASPECIFIC"`.

nom_status Nomenclatural status (may contain multiple values).

bracket_authorship Basionym author in parentheses.

bracket_year Basionym author year.

gbif_year Combining author year.

name_published_in Publication citation.

origin How the name entered the backbone.

infra_specific_epithet Infraspecific epithet.

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify("Quercus robur", backbone = "gbif") |>
  add_gbif_info()

options(old)
```

`add_gidias` *Add invasive-species impact (EICAT / SEICAT, GIDIAS)*

Description

Joins per-species environmental- and socio-economic-impact aggregates from GIDIAS (Bacher et al. 2025), the IPBES invasive-species assessment's global impact compilation, to a `taxify()` result by looking up `accepted_name`. GIDIAS's individual impact records are reduced to per-species indicators; the raw impact records are not distributed.

Usage

```
add_gidias(x, group = "Any", cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`group`	Character. Which affected native taxon the impact is measured against: • `"Any"` (default): every impact record for the species, the species-level summary. • One of `"Plant"`, `"Invertebrate"`, `"Vertebrate"`, `"Microbe"`, `"Fungi"`: only the impacts recorded against that group. • Several (e.g. `c("Plant", "Vertebrate")`): adds columns with a group suffix (e.g. `gidias_eicat_category_Plant`). • `"all"`: one column set per group, `"Any"` included. List them with `enrichment_groups("gidias")`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

A species' EICAT category is the most severe magnitude among its negative (harmful) environmental-impact records, on the IUCN scale `"MC"` (Minimal Concern), `"MN"` (Minor), `"MO"` (Moderate), `"MR"` (Major, local extinction), `"MV"` (Massive, global extinction), or `"DD"` (Data Deficient); `gidias_seicat_category` applies the same rule to socio-economic impact (SEICAT, `"MC"` to `"MR"`). For impact reconciled across sources, use `add_trait()` with `"environmental_impact"` or `"socioeconomic_impact"`.

Source: GIDIAS (Bacher et al. 2025, Scientific Data; CC BY 4.0), compiled for the IPBES thematic assessment report on invasive alien species. Only the derived per-species aggregates are distributed here, not the raw impact records.

Value

The same data.frame with additional columns. The default set:

gidias_eicat_category EICAT environmental-impact category (MC/MN/MO/MR/MV/DD, or NA when no negative impact is recorded).

gidias_eicat_mechanism IUCN EICAT impact mechanism(s) behind the species' negative environmental impacts.

gidias_seicat_category SEICAT socio-economic-impact category (MC/MN/MO/MR/DD, or NA).

gidias_ncp_direction Direction of the species' impact on nature's contributions to people: "Negative", "Positive", "Neutral", or a combination ("Negative; Positive") where the records disagree. GIDIAS scores no magnitude for this block, so there is no category to report.

gidias_ias_taxon Functional group: Plant, Invertebrate, Vertebrate, or Microbe.

gidias_realms Realm(s) the impacts span: terrestrial, freshwater, and/or marine.

gidias_n_records Number of impact records for the species.

gidias_n_sources Number of distinct sources documenting the impacts.

cols = "all" additionally attaches the numeric EICAT/SEICAT magnitudes (0-3), the affected well-being constituents and contributions to people (gidias_ncp_affected), kingdom, the negative-record count, and a global-extinction flag.

What a species impacts

GIDIAS records the affected native taxon per impact record, so group asks the question at that grain: add_gidias(group = "Invertebrate") scores *Felis catus* "MO" (reduced populations) where the default "Any" scores it "MV", the global extinctions it drove among vertebrates. 28% of species with a recorded affected taxon impact two or more of the five groups, so for those the species-level category is the most severe impact on anything, not the impact on each thing.

The vocabulary is the coarse one GIDIAS controls: "Vertebrate", not "Aves". "Any" is the only group carrying the impact records with no affected taxon recorded (12% of the negative ones) and the two people-facing blocks, so gidias_seicat_category and gidias_ncp_direction are NA on every other group: impact on people is not a question the affected-native-taxon axis can answer.

References

Bacher S et al. (2025) Global Impacts Dataset of Invasive Alien Species (GIDIAS). Scientific Data 12:832. doi:10.1038/s41597025051845

Examples

```
## Not run:
# Downloads the GIDIAS enrichment on first use.
taxify("Felis catus", backbone = "gbif") |>
  add_gidias()

# What the cat does to invertebrates, rather than to anything at all.
taxify("Felis catus", backbone = "gbif") |>
  add_gidias(group = "Invertebrate")
```

```
taxify("Felis catus", backbone = "gbif") |>
  add_gidias(group = c("Invertebrate", "Vertebrate"))

## End(Not run)
```

---

add_gift *Add plant traits from GIFT*

Description

Joins species-level plant traits from GIFT, the Global Inventory of Floras and Traits (Weigelt et al. 2020), to a `taxify()` result by `accepted_name`. GIFT aggregates published trait records to one value per species (mean for numeric traits, most frequent entry for categorical ones). `cols` chooses which traits to attach; browse the available columns with `gift_traits()`.

Usage

```
add_gift(x, cols = NULL, verbose = TRUE)
```

Arguments

x	A data.frame returned by `taxify()`.
cols	Which GIFT trait columns to attach. One of: `NULL` (the default) for a convenient set of well-populated traits; the string `"all"` for every bundled trait; or a character vector of `gift_` column names (e.g. `"plant_height_max"`, with or without the `gift_` prefix). See `gift_traits()`. When left `NULL`, a one-time message notes the default set and how to request all traits.
verbose	Logical. Default `TRUE`.

Details

The GIFT trait table is bundled as a pre-built `.vtr` and joined offline, so no internet access is needed once it is present (the first use downloads it, or builds it from source if `taxifydb` is installed). GIFT's API exposes only the redistributable subset of its data (CC BY 4.0; references whose underlying source is restricted are excluded), and that subset is what is bundled here. Cite GIFT and, where applicable, the underlying references (`GIFT::GIFT_references()`) for the values used.

Value

The same data.frame with one added column per requested trait, named `gift_<trait>`. Numeric traits (heights, masses, areas) are doubles, the rest character. Rows with no value in GIFT get `NA`. With the default `cols`, the added columns are `gift_woodiness_1`, `gift_growth_form_1`, `gift_lifecycle_1`, `gift_life_form_1`, `gift_climber_1`, `gift_epiphyte_1`, `gift_parasite_1`, `gift_aquatic_1`, `gift_plant_height_max`, `gift_photosynthetic_pathway`, `gift_seed_mass_mean`, `gift_dispersal_syndrome_1`, `gift_flowering_start`, `gift_flowering_end`, `gift_deciduousness_1`, and `gift_sla_mean`.

References

Weigelt P, Konig C, Kreft H (2020) GIFT - A Global Inventory of Floras and Traits for macroecology and biogeography. Journal of Biogeography 47:16-43. doi:10.1111/jbi.13623 Denelle P, Weigelt P, Kreft H (2023) GIFT: an R package to access the Global Inventory of Floras and Traits. Methods in Ecology and Evolution 14:2738-2748. doi:10.1111/2041210X.14213

See Also

`gift_traits()` to browse the available columns.

Examples

```
old <- options(taxify.data_dir = taxify_example_data())

taxify("Abies alba") |>
  add_gift()

options(old)
```

---

add_globi *Add biotic interaction degree (GloBI)*

Description

Joins per-species biotic interaction breadth from GloBI (Global Biotic Interactions) to a `taxify()` result by looking up `accepted_name`. GloBI's aggregated interaction records are reduced to per-species counts: how many distinct partner taxa a species interacts with (undirected), across how many distinct interaction types, over how many interaction records.

Usage

```
add_globi(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: GloBI (Poelen et al. 2014), an open index of biotic interactions aggregated from many contributed datasets. Only derived per-species counts are distributed here; the underlying interaction records carry the licenses of their original data contributors, who should be cited in derivative work. Partner counts are resolved to accepted names before counting, so synonymous partners are not double-counted.

Value

The same data.frame with additional columns:

interaction_degree Number of distinct partner taxa recorded interacting with the species (both directions).

n_interaction_types Number of distinct interaction types (eats, pollinates, parasitises, ...).

n_interaction_records Total number of interaction records touching the species.

References

Poelen JH, Simons JD, Mungall CJ (2014) Global Biotic Interactions: An open infrastructure to share and analyze species-interaction datasets. Ecological Informatics 24:148-159. doi:10.1016/j.ecoinf.2014.08.005

Examples

```
## Not run:
taxify("Apis mellifera", backbone = "gbif") |>
  add_globi()

## End(Not run)
```

---

add_globtherm *Add thermal tolerance limits (GlobTherm)*

Description

Joins GlobTherm upper and lower thermal tolerance limits to a `taxify()` result by looking up `accepted_name`.

Usage

```
add_globtherm(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: GlobTherm (Bennett et al. 2018, Scientific Data, CC0). Coverage: ~2.1k species across aquatic and terrestrial groups.

Value

The same data.frame with additional columns:

globtherm_thermal_max_c Upper thermal limit (degrees Celsius).

globtherm_thermal_max_metric Definition of the upper limit (e.g. ctmax, LT50, UTNZ); the value is ambiguous without it.

globtherm_thermal_min_c Lower thermal limit (degrees Celsius).

globtherm_thermal_min_metric Definition of the lower limit (e.g. ctmin, LT50, LTNZ).

globtherm_thermal_max_error Reported error on the upper limit.

globtherm_thermal_min_error Reported error on the lower limit.

References

Bennett JM et al. (2018) GlobTherm, a database on the thermal tolerance for aquatic and terrestrial organisms. Scientific Data 5:180022. doi:10.1038/sdata.2018.22

Examples

```
## Not run:
taxify("Lepomis gibbosus", backbone = "gbif") |>
  add_globtherm()

## End(Not run)
```

---

add_glonaf *Add naturalized alien flora status (GloNAF)*

Description

Joins GloNAF (Global Naturalized Alien Flora) data to a `taxify()` result, filtered by region.

Usage

```
add_glonaf(x, region, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`region`	Character. GloNAF region identifier(s), or `"all"`. Regions are ISO 3166 alpha-3 country codes extended with dot notation for sub-national units (e.g., `"DEU"` for Germany, `"USA.CA"` for California). List them with `enrichment_groups("glonaf")`. • Single region: adds `naturalized` column (no suffix). • Multiple regions: adds `naturalized_<region>` columns. • `"all"`: adds one column per region in the dataset.
`verbose`	Logical. Default `TRUE`.

Details

Source: GloNAF v2.0 (van Kleunen et al. 2019, Davis et al. 2025, CC BY 4.0). Coverage: ~16k alien plant taxa across ~1,300 regions. Plants only.

Value

The same data.frame with additional column(s):

naturalized 1 if the species is recorded as naturalized in that region, NA otherwise.

References

van Kleunen M et al. (2019) The Global Naturalized Alien Flora (GloNAF) database. Ecology 100:e02542. doi:10.1002/ecy.2542

Davis AJS et al. (2025) The updated Global Naturalized Alien Flora (GloNAF 2.0) database. Ecology 106:e70245. doi:10.1002/ecy.70245

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify("Robinia pseudoacacia") |>
  add_glonaf(region = "DEU")

taxify("Robinia pseudoacacia") |>
  add_glonaf(region = c("DEU", "AUT"))

options(old)
```

---

add_gmpd *Add mammal parasite burden (GMPD 2.0)*

Description

Joins host-level parasite summaries from the Global Mammal Parasite Database (Stephens et al. 2017) to a `taxify()` result by looking up `accepted_name`. The database's host-parasite association records are aggregated to one row per host species.

Usage

```
add_gmpd(x, cols = NULL, verbose = TRUE)
```

Arguments

x	A data.frame returned by taxify().
cols	Which columns to attach: NULL (default) the curated set, "all" every column the source carries, or a character vector of names. See enrichment_cols.
verbose	Logical. Default TRUE.

Details

Covers wild carnivores, ungulates and primates (462 host species).

Parasite richness is a sampling-sensitive count: a well-studied host accumulates more recorded parasites than a rarely sampled one, so the values reflect research effort as well as biology.

This source states no licence. It is distributed under the journal's version-of-record terms with a citation request, and taxify ships no pre-built copy of it. The first call builds it from the original source on the user's machine, which requires the taxifydb package (`remotes::install_github("gcol33/taxifydb")`). taxify redistributes none of the data. Cite Stephens et al. (2017) for the data.

Value

The same data.frame with additional columns. The default set:

gmpd_parasite_richness Distinct parasite species recorded for the host.

gmpd_n_helminth, gmpd_n_virus, gmpd_n_bacteria, gmpd_n_protozoa, gmpd_n_arthropod, gmpd_n_fungus, gmpd
Distinct parasite species by parasite type.

gmpd_mean_prevalence Mean reported prevalence across the host's records.

gmpd_host_group Host group (carnivores, ungulates, primates).

References

Stephens PR, Pappalardo P, Huang S, et al. (2017) Global Mammal Parasite Database version 2.0. Ecology 98:1476. doi:10.1002/ecy.1799

Examples

```
## Not run:
# Builds the enrichment on first use (needs taxifydb).
taxify("Panthera leo", backbone = "gbif") |>
  add_gmpd()

## End(Not run)
```

`add_griis` *Add invasive species status (GRIIS)*

Description

Joins GRIIS (Global Register of Introduced and Invasive Species) status to a `taxify()` result, filtered by country. This function is named after its source, GRIIS; GloNAF (`add_glonaf()`) carries related naturalized-alien status.

Usage

```
add_griis(x, country, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`country`	Character. ISO 3166-1 alpha-2 country code(s), or `"all"`. • Single code (e.g., `"AT"`): adds `invasive_status` column (no suffix). • Multiple codes (e.g., `c("AT", "DE")`): adds `invasive_status_AT`, `invasive_status_DE`. • `"all"`: adds one column per country in the dataset. List the codes GRIIS covers with `enrichment_groups("griis")`.
`cols`	Which columns to attach. `NULL` (the default) attaches the curated set; a character vector of column names attaches just those, and `"all"` attaches every column the source carries (see `enrichment_cols()`).
`verbose`	Logical. Default `TRUE`.

Details

Source: GRIIS (Zenodo combined CSV, CC BY 4.0, 196 countries). Coverage: ~23k name x country combinations.

Value

The same data.frame with additional column(s):

invasive_status One of `"native"`, `"introduced"`, `"invasive"`, or `NA` if not recorded for that country.

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify("Robinia pseudoacacia") |>
  add_griis(country = "AT")

taxify("Robinia pseudoacacia") |>
```

```
  add_griis(country = c("AT", "DE"))

options(old)
```

`add_groot` *Add root traits (GRooT)*

Description

Joins species-level root traits from the Global Root Traits (GRooT) database to a `taxify()` result by looking up `accepted_name`. GRooT aggregates root trait records to per-species means. The `.vtr` carries the full GRooT trait set (38 root traits); the default attaches the nine best-populated key traits, and `cols = "all"` attaches every one. Run `enrichment_cols("groot")` to list them.

Usage

```
add_groot(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the nine key traits below, `"all"` every GRooT trait, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: GRooT database (Guerrero-Ramirez et al. 2021). Vascular plants. GRooT data are publicly available and used here with the data-paper citation requested by the authors.

Value

The same data.frame with additional columns (per-species means). The default set:

root_diameter Mean root diameter.

specific_root_length Specific root length.

root_tissue_density Root tissue density.

root_n_concentration Root nitrogen concentration.

root_c_concentration Root carbon concentration.

root_mass_fraction Root mass fraction.

lateral_spread Lateral spread.

root_mycorrhizal_colonization Root mycorrhizal colonization intensity.

rooting_depth Maximum rooting depth.

`cols = "all"` additionally attaches root chemistry (P/K/Ca/Mg/Mn concentrations, C:N and N:P ratios), architecture (branching density and ratio, stele diameter and fraction, cortex thickness, vessel diameter and number), turnover (root lifespan, production, turnover rate, litter mass-loss rate), and specific root area, respiration, and dry-matter content, among others. Units follow the GRooT data paper; see the reference below.

References

Guerrero-Ramirez NR et al. (2021) Global root traits (GRooT) database. Global Ecology and Biogeography 30:25-37. doi:10.1111/geb.13179

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify("Abies alba") |>
  add_groot()

options(old)
```

---

add_gwdd *Add wood density (Global Wood Density Database v2)*

Description

Joins species-level wood density to a `taxify()` result by looking up `accepted_name`. Wood density is reported as wood specific gravity (oven-dry mass / green volume), dimensionless and numerically equal to g/cm3.

Usage

```
add_gwdd(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Global Wood Density Database v2 (Fischer et al. 2026, New Phytologist, CC BY 4.0). Coverage: ~17.3k species. Bark density is not part of the aggregated source and is not included.

Value

The same data.frame with additional columns:

gwdd_wood_density_g_cm3 Species-mean wood density (g/cm3).

gwdd_wood_density_trunk_g_cm3 Trunk wood density (g/cm3).

gwdd_wood_density_branch_g_cm3 Branch wood density (g/cm3).

gwdd_n_measurements Number of underlying measurements.

References

Fischer FJ et al. (2026) The Global Wood Density Database version 2. New Phytologist. doi:10.1111/nph.70860

Examples

```
## Not run:
taxify("Quercus robur", backbone = "gbif") |>
  add_gwdd()

## End(Not run)
```

---

add_homerange *Add mammal home-range size (HomeRange)*

Description

Joins species-median home-range size and body mass to a `taxify()` result by `accepted_name`.

Usage

```
add_homerange(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Broekman et al. (2023) HomeRange database (Dryad, CC0). Per-individual records are reduced to species medians.

Value

The same data.frame with numeric `homerange_home_range_km2` and `homerange_body_mass_kg`.

References

Broekman MJE et al. (2023) HomeRange: a global database of mammalian home ranges. Dryad. doi:10.5061/dryad.d2547d85x

Examples

```
## Not run:
taxify("Panthera leo", backbone = "gbif") |>
  add_homerange()

## End(Not run)
```

---

add_hosts *Add Lepidoptera hostplant breadth (NHM HOSTS)*

Description

Joins per-insect hostplant breadth from the Natural History Museum HOSTS database to a `taxify()` result by looking up `accepted_name`. Each moth or butterfly species is summarised by how many distinct hostplants and hostplant families it has been recorded feeding on.

Usage

```
add_hosts(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Robinson et al. (2010) HOSTS, Natural History Museum, London (CC0). Lepidoptera only; ~24k species with at least one recorded hostplant.

Value

The same data.frame with additional columns:

host_plant_count Number of distinct hostplant species recorded.

host_family_count Number of distinct hostplant families recorded.

References

Robinson GS, Ackery PR, Kitching IJ, Beccaloni GW, Hernandez LM (2010) HOSTS - a Database of the World's Lepidopteran Hostplants. Natural History Museum, London. doi:10.5519/havt50xw

Examples

```
## Not run:
taxify("Papilio machaon", backbone = "gbif") |>
  add_hosts()

## End(Not run)
```

---

add_huang_amph *Add amphibian morphometrics (Huang)*

Description

Joins species-level amphibian body measurements to a `taxify()` result by `accepted_name`. Only measurements comparable across Anura, Caudata and Gymnophiona are carried; per-specimen values are reduced to species medians.

Usage

```
add_huang_amph(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Huang amphibian morphological dataset (figshare, CC-BY 4.0).

Value

The same data.frame with numeric `huang_amph_svl_mm`, `huang_amph_head_length_mm`, `huang_amph_head_width_mm`, `huang_amph_eye_diameter_mm`, `huang_amph_forelimb_length_mm`, `huang_amph_hindlimb_length_mm` and categorical `huang_amph_taxon_order`.

References

Huang et al. A global amphibian morphological trait dataset. figshare. doi:10.6084/m9.figshare.21159229

Examples

```
## Not run:
taxify("Bufo bufo", backbone = "gbif") |>
  add_huang_amph()

## End(Not run)
```

`add_hybrid_info` *Add hybrid parent and type information*

Description

Parses the `input_name` column from a `taxify()` result to extract hybrid parent names and classify the hybrid type.

Usage

```
add_hybrid_info(x)
```

Arguments

`x`	A data.frame returned by `taxify()`.

Value

The same data.frame with additional columns:

hybrid_parent_1 First parent (full binomial), `NA` if not a hybrid formula.

hybrid_parent_2 Second parent (full binomial, abbreviated or omitted genus expanded), `NA` if not a hybrid formula.

hybrid_parent_1_accepted, hybrid_parent_2_accepted The accepted name each parent resolves to against the backbone(s) used for `x` (from the result's metadata), or `NA` if the parent did not match.

hybrid_parent_1_id, hybrid_parent_2_id The backbone `taxon_id` of the accepted taxon each parent resolves to, usable directly with `id2name()`, or `NA` if the parent did not match. An unresolved hybrid formula carries `NA` in the result's own `taxon_id` (it has no single backbone record); these two columns are where the component IDs live.

hybrid_type One of `"nothogenus"`, `"nothospecies"`, `"formula"`, or `NA` if not a hybrid (same value as the `hybrid_type` column already on a `taxify()` result).

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify("Quercus pyrenaica x Q. petraea") |>
  add_hybrid_info()

options(old)
```

`add_hydraulics` *Add plant hydraulic traits (Sanchez-Martinez et al.)*

Description

Joins the drought-mortality trait block (xylem embolism resistance, xylem and leaf conductivity, sapwood-to-leaf allocation and drought exposure) to a `taxify()` result by accepted name.

Usage

```
add_hydraulics(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) all, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Sanchez-Martinez et al. (2020), 2024 seed-plant species. CC BY 4.0. The widest P50 coverage in the bundled sources by an order of magnitude: 894 species against the 123 AusTraits and BROT carry between them.

This is a literature compilation and so are AusTraits and BROT, so where they overlap they often carry the same primary record: 63% of the P50 values shared with AusTraits and 65% of the Ks values are equal to the last digit. Agreement between them is not independent corroboration. The overlap is small against what this adds: 74 of its 894 P50 species appear in either incumbent.

`huber_value_cm2_m2` is square centimetres of sapwood per square metre of leaf, so it runs 1e4 above the dimensionless AusTraits column. The factor is measured rather than assumed: 79 of the 208 species the two share are equal to the last digit once converted. `add_trait()` applies it.

`mean_annual_temp_c` describes where the species grows, not what it tolerates; it feeds `climatic_temp_mean`, never `thermal_max`.

Value

The same data.frame with hyd_ columns: p50_mpa, sapwood_conductivity, leaf_conductivity, huber_value_cm2_m2, min_water_potential_mpa, hydraulic_safety_margin_mpa, mean_annual_temp_c.

References

Sanchez-Martinez P, Martinez-Vilalta J, Dexter KG, Segovia RA, Mencuccini M (2020) Adaptation and coordinated evolution of plant hydraulic traits. Ecology Letters 23:1599-1610. doi:10.1111/ele.13584

Data: doi:10.6084/m9.figshare.12625418.v1

Examples

```
## Not run:
taxify(c("Quercus ilex", "Abies alba")) |>
  add_hydraulics()

## End(Not run)
```

---

add_imageomics_neon *Add North American ground-beetle elytra measurements (Imageomics / NEON)*

Description

Joins per-species elytra measurements, taken from images of pinned NEON specimens, to a taxify() result by accepted name.

Usage

```
add_imageomics_neon(x, cols = NULL, verbose = TRUE)
```

Arguments

x	A data.frame returned by taxify().
cols	Which columns to attach: NULL (default) all of them, or a character vector of names. See enrichment_cols.
verbose	Logical. Default TRUE.

Details

Source: the Imageomics `2018-NEON-beetles` dataset, 39,064 measurements of pinned individuals across 30 NEON sites, reduced to a per-species median over 75 named species.

This is the only North American ground-beetle morphometry taxify carries, and the only carabid measurement source independent of carabids.org, whose values every European carabid table inherits.

The source measures the elytron rather than the whole animal, in centimetres; values here are millimetres. Measurements of specimens photographed at an angle (`lying_flat = "No"`) are excluded, since a projected length is foreshortened. The length-to-width ratio has a median of 1.79 across species, the shape of a carabid elytron, and *Carabus nemoralis* reads 14.87 mm against the 23.4 mm body length that `add_finand()` measured on European specimens.

Value

The same data.frame with `neon_` columns: numeric `elytra_length_mm`, `elytra_width_mm`, and `measurement_n` (the number of individual measurements behind the median).

References

Imageomics Institute. 2018-NEON-beetles. Hugging Face. https://huggingface.co/datasets/imageomics/2018-NEON-beetles

Examples

```
## Not run:
taxify(c("Carabus nemoralis", "Pasimachus californicus")) |>
  add_imageomics_neon()

## End(Not run)
```

---

add_inrae_collembola_traits

Add Collembola traits (Data INRAE deposits)

Description

Joins fuzzy-coded springtail functional traits to a `taxify()` result by looking up `accepted_name`. Output columns are prefixed `inrae_`.

Usage

```
add_inrae_collembola_traits(x, cols = NULL, verbose = TRUE)
```

Arguments

x	A data.frame returned by taxify().
cols	Which columns to attach: NULL (default) all, or a character vector of names. See enrichment_cols.
verbose	Logical. Default TRUE.

Details

Source: fuzzy-coded Collembola traits compiled from the BETSI database (Pey et al. 2014) across two Data INRAE deposits, the datasets behind Joimel et al. (2021). Coverage: 135 species. The deposits' species codes carry no published legend and are decoded against a Collembola reference pool; codes that cannot be resolved are dropped, never guessed. The fuzzy affinity vectors are not passed to the scalar add_trait() verb.

Value

The same data.frame with additional columns, one per modality bin of seven fuzzy-coded traits: number of ocelli, furca, post-antennal organ, pigmentation, body shape, scales and reproduction. Each trait is fuzzy coded, so a species' affinities across the bins of one trait (columns sharing an inrae_<trait>__ stem) sum to 100. The source is sparse, so a trait a species was not scored for is NA across its bins. See enrichment_cols for the full column list.

References

Joimel S et al. (2021) Collembola are among the most flexible soil fauna: a comparison across land uses. Frontiers in Ecology and Evolution 9:630919. doi:10.3389/fevo.2021.630919

Examples

```
## Not run:
taxify("Isotoma viridis", backbone = "gbif") |>
  add_inrae_collembola_traits()

## End(Not run)
```

---

add_invacost *Add economic cost of biological invasions (InvaCost)*

Description

Joins per-species economic-cost aggregates from InvaCost (Diagne et al. 2020) to a taxify() result by looking up accepted_name. InvaCost's individual cost estimates are reduced to three per-species indicators; the raw estimate rows are not distributed.

Usage

```
add_invacost(x, cols = NULL, verbose = TRUE)
```

Arguments

x	A data.frame returned by taxify().
cols	Which columns to attach: NULL (default) the curated set, "all" every column the source carries, or a character vector of names. See enrichment_cols.
verbose	Logical. Default TRUE.

Details

Source: InvaCost (Diagne et al. 2020, Scientific Data; database version 4.1, CC BY 4.0). Only the derived per-species aggregates are distributed here. For a rigorous cost analysis (temporal expansion, overlap handling, reliability filtering) use the invacost R package on the full database.

Value

The same data.frame with additional columns:

invacost_cost_total_usd Cumulative documented cost in 2017 USD: each estimate's standardised annual cost expanded across its documented impact period and summed over all of the species' estimates. Overlapping estimates are not de-duplicated, so this is a coarse economic-impact indicator, not an audited figure.

invacost_cost_n Number of cost estimates recorded for the species.

invacost_cost_type Cost type carrying the largest share of the species' cumulative cost: damage, management, mixed, or unspecified.

References

Diagne C et al. (2020) InvaCost, a public database of the economic costs of biological invasions worldwide. Scientific Data 7:277. doi:10.1038/s4159702000586z

Examples

```
## Not run:
taxify("Solenopsis invicta", backbone = "gbif") |>
  add_invacost()

## End(Not run)
```

---

add_italic *Add Italian-lichen taxon-page traits (ITALIC)*

Description

Joins per-species morphological and ecological descriptors from ITALIC, the Information System on Italian Lichens, to a taxify() result by looking up accepted_name. One row per species, scraped from the ITALIC 8.0 taxon pages.

Usage

```
add_italic(x, cols = NULL, verbose = TRUE)
```

Arguments

x	A data.frame returned by taxify().
cols	Which columns to attach: NULL (default) or "all" attaches every column the source carries, or a character vector of names. See enrichment_cols.
verbose	Logical. Default TRUE.

Details

Source: ITALIC 8.0 (Nimis; Univ. of Trieste), taxon-page descriptors, CC BY-SA 4.0. Lichens are otherwise almost absent from the bundled trait databases.

Value

The same data.frame with additional columns:

growth_form Thallus growth form (crustose, foliose, fruticose, ...).

substrata Substrata the species grows on.

photobiont Photosynthetic partner.

reproductive_strategy Reproductive strategy.

References

Nimis PL. ITALIC - The Information System on Italian Lichens, Version 8.0. University of Trieste, Dept. of Biology (https://italic.units.it), accessed 2026-07. System paper: Martellos S, Conti M, Nimis PL (2023) Aggregation of Italian Lichen Data in ITALIC 7.0. Journal of Fungi 9(5):556. doi:10.3390/jof9050556

Examples

```
## Not run:
taxify("Xanthoria parietina", backbone = "gbif") |>
  add_italic()

## End(Not run)
```

add_iucn *Add IUCN Red List conservation status*

Description

Joins IUCN Red List conservation status to a `taxify()` result by looking up `accepted_name`. This function is named after its source, the IUCN Red List; for conservation status reconciled across sources, use `add_trait()` once more than one source is registered for it.

Usage

```
add_iucn(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Show download progress if enrichment data needs to be fetched. Default `TRUE`.

Details

Conservation status values are compiled from publicly available sources including GBIF and the IUCN Red List API. Coverage is global across all taxonomic groups (~166k species).

Value

The same data.frame with an additional column:

conservation_status IUCN category: `"LC"` (Least Concern), `"NT"` (Near Threatened), `"VU"` (Vulnerable), `"EN"` (Endangered), `"CR"` (Critically Endangered), `"EW"` (Extinct in the Wild), `"EX"` (Extinct), or `NA` if not assessed.

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify("Panthera tigris", backbone = "gbif") |>
  add_iucn()

options(old)
```

`add_kew_cvalues` *Add plant genome size (Kew Plant DNA C-values)*

Description

Joins plant genome-size data from the Kew Plant DNA C-values database (Pellicer & Leitch 2020) to a `taxify()` result by looking up `accepted_name`. The 1C DNA amount is a per-species median. The chromosome number and ploidy level are the base cytotype, with the range across a species' cytotypes in the `_min` / `_max` columns.

Usage

```
add_kew_cvalues(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Kew Plant DNA C-values database, release 7.1 (Royal Botanic Gardens Kew), CC BY. Vascular plants.

Value

The same data.frame with additional columns. The default set:

cval_genome_size_1c_pg Genome size (1C DNA amount, picograms).

cval_chromosome_2n Somatic chromosome number (2n), base cytotype.

cval_ploidy_x Ploidy level, base cytotype.

`cols = "all"` also attaches the per-species min/max/n spread of each value, plus per-species provenance:

cval_original_reference The paper(s) the values were measured in, joined by `"; "` where a species carries records from several.

cval_estimation_method The method(s) used, in the source's codes: `FC:PI` and other `FC:` variants are flow cytometry with the named stain, `Fe` is Feulgen densitometry. The two are not interchangeable, so a species measured by both reports both.

References

Pellicer J, Leitch IJ (2020) The Plant DNA C-values database (release 7.1): an updated online repository of plant genome size data for comparative studies. New Phytologist 226:301-305.

Examples

```
## Not run:
# Downloads the enrichment on first use.
taxify("Zea mays", backbone = "gbif") |>
  add_kew_cvalues()

## End(Not run)
```

`add_kew_sid` *Add seed traits from the Kew Seed Information Database (SER-SID)*

Description

Joins species-level seed traits from the Kew Seed Information Database (SER-SID) to a `taxify()` result by looking up `accepted_name`.

Usage

```
add_kew_sid(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Royal Botanic Gardens Kew, Seed Information Database, served as SER-SID (https://ser-sid.org/), CC BY 2.0. Per-record measurements are reduced to per-species medians (numeric) and modes (categorical); a thousand-seed weight in grams equals the per-seed mass in milligrams.

Value

The same data.frame with additional columns. The curated set:

sid_thousand_seed_weight Thousand-seed weight (grams per 1000 seeds, median of all records).

sid_storage_behaviour Seed storage behaviour (Orthodox/Recalcitrant/Intermediate/Uncertain).

sid_oil_content_pct Seed oil content (percent, median).

sid_protein_content_pct Seed protein content (percent, median).

sid_lifeform Raunkiaer life-form code as recorded by SID.

With `cols = "all"` the seed-weight record count (`n_seed_weight_records`) and modal `fruit_type` are also attached under their source names. Joined on `accepted_name`.

References

Royal Botanic Gardens Kew. Seed Information Database (SID). https://ser-sid.org/

Examples

```
old <- options(taxify.data_dir = taxify_example_data())

taxify("Quercus robur", backbone = "gbif") |>
  add_kew_sid()

options(old)
```

`add_leda` *Add plant traits from LEDA Traitbase*

Description

Joins LEDA Traitbase (Kleyer et al. 2008) plant functional traits to a `taxify()` result by looking up `accepted_name`. LEDA provides species-level trait data for NW European plant species, covering life form, dispersal, seed, leaf, and clonality traits.

Usage

```
add_leda(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: LEDA Traitbase (Kleyer et al. 2008). Coverage: ~12,500 NW European plant taxa. `cols = "all"` adds the remaining LEDA traits and, for each, a `<col>_source` column of the references behind the value (resolve with `cite()` and `source = "leda"`).

The Raunkiaer life form is a bud-position classification system: phanerophyte = buds >25 cm above soil, chamaephyte = buds near soil surface, hemicryptophyte = buds at soil surface, geophyte (cryptophyte) = buds below soil, therophyte = annual that survives as seed.

Value

The same data.frame with additional columns:

raunkiaer_life_form Most frequent LEDA plant growth form: the Raunkiaer classes (Phanerophyte, Chamaephyte, Hemicryptophyte, Geophyte, Therophyte, Hydrophyte) plus Liana, Vascular semi-parasite and Vascular parasite.

raunkiaer_variable 1 if the species' records name more than one growth form, 0 otherwise.

dispersal_type Most frequent dispersal type, as LEDA's -chor term (meteorochor, epizoochor, nautochor, ...).

terminal_velocity_ms Diaspore terminal velocity in m/s (species median).

seed_mass_mg Seed mass in mg (species median). Prefixed with `leda_` in the .vtr to avoid collision with Diaz traits.

canopy_height_m Canopy height in metres (species median): the height of the highest photosynthetic tissue, which LEDA distinguishes from plant height.

leaf_mass_mg Leaf dry mass in mg (species median).

sla_mm2_mg Specific leaf area in mm^2/mg (species median).

clonal_growth_organ Most frequent primary clonal growth organ (epigeogeneous stems, root-splitters, bulbs, ...).

floating_capacity_1week_pct Percentage of diaspores still floating after one week (species median).

References

Kleyer M et al. (2008) The LEDA Traitbase: a database of life-history traits of the Northwest European flora. Journal of Ecology 96:1266-1274.

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify("Arrhenatherum elatius") |>
  add_leda()

options(old)
```

`add_leptraits` *Add butterfly traits (LepTraits)*

Description

Joins LepTraits 1.0 butterfly life-history and ecological traits to a `taxify()` result by looking up `accepted_name`.

Usage

```
add_leptraits(x, cols = NULL, verbose = TRUE)
```

Arguments

x	A data.frame returned by taxify().
cols	Which columns to attach: NULL (default) the curated set, "all" every column the source carries, or a character vector of names. See enrichment_cols.
verbose	Logical. Default TRUE.

Details

Source: LepTraits 1.0 (Shirey et al. 2022, CC0). Coverage: ~12.4k butterfly species globally (Papilionoidea).

Value

The same data.frame with additional columns:

wingspan_mm Wingspan in mm (midpoint of lower and upper bounds).

voltinism Number of generations per year.

diapause_stage Overwintering/diapause life stage.

canopy_affinity Canopy association category.

edge_affinity Edge/gap affinity category.

moisture_affinity Moisture affinity category.

disturbance_affinity Disturbance affinity category.

n_hostplant_families Number of host plant families used.

flight_months Number of months with adult flight activity.

References

Shirey V et al. (2022) LepTraits 1.0: A globally comprehensive dataset of butterfly traits. Scientific Data 9:398.

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify("Vanessa cardui", backbone = "gbif") |>
  add_leptraits()

options(old)
```

`add_madin` *Add bacterial and archaeal traits (Madin et al.)*

Description

Joins species-level bacterial and archaeal phenotypic and genome traits to a `taxify()` result by looking up `accepted_name`.

Usage

```
add_madin(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Madin et al. (2020, Scientific Data, CC BY 4.0). Coverage: ~14.9k bacterial and archaeal species.

Value

The same data.frame with additional columns:

madin_gram_stain Gram stain (positive/negative).

madin_metabolism Metabolism (aerobic/anaerobic/facultative/...).

madin_cell_shape Cell shape (bacillus/coccus/spiral/...).

madin_motility Motility (yes/no/flagella/gliding/...).

madin_sporulation Sporulation (yes/no).

madin_isolation_source Isolation source category.

madin_growth_temp_c Recorded growth temperature (degrees Celsius).

madin_optimum_temp_c Optimum growth temperature (degrees Celsius).

madin_optimum_ph Optimum growth pH.

madin_genome_size_bp Genome size (base pairs).

madin_gc_content_pct Genomic G+C content (percent).

References

Madin JS et al. (2020) A synthesis of bacterial and archaeal phenotypic trait data. Scientific Data 7:170. doi:10.1038/s4159702004974

Examples

```
## Not run:
taxify("Escherichia coli", backbone = "gbif") |>
  add_madin()

## End(Not run)
```

`add_monograph_collembola_body_length` *Add Collembola body length (monographs)*

Description

Joins per-species springtail body length to a `taxify()` result by looking up `accepted_name`. The value is mined from three primary Collembola monographs and aggregated per species: the median of the per-monograph medians, with the observed range and the measurement and monograph counts.

Usage

```
add_monograph_collembola_body_length(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) all, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: body length mined from three primary Collembola monographs, Stach (1957, Neelidae and Dicyrtomidae), Hopkin (2007, Britain and Ireland) and Bretfeld (1999, Symphypleona). The source monographs are copyright their publishers; only the extracted measurements are redistributed, for non-commercial scientific use. Coverage: 481 species from 679 measurements. A re-extraction independent of BETSI, more complete than BETSI on the shared books (210 vs 145 species from Bretfeld 1999) and reaching Stach's Neelidae and Dicyrtomidae, which fall outside BETSI's Collembola coverage. Body length is also available through `add_trait()("body_length")`.

Value

The same data.frame with additional columns:

monograph_body_length_mm Body length (mm), median of the per-monograph medians.

monograph_body_length_min_mm Smallest measurement (mm).

monograph_body_length_max_mm Largest measurement (mm).

monograph_body_length_n Number of measurements aggregated.

monograph_body_length_sources Number of monographs contributing.

References

Hopkin SP (2007) A Key to the Collembola (Springtails) of Britain and Ireland. FSC Publications, Shrewsbury.

Examples

```
## Not run:
taxify("Isotoma viridis", backbone = "gbif") |>
  add_monograph_collembola_body_length()

## End(Not run)
```

---

add_nesttrait *Add bird nest traits (NestTrait)*

Description

Joins bird nest-site, nest-structure and nest-attachment indicators to a `taxify()` result by `accepted_name`. Each trait is a 0/1 presence flag; a species may carry several flags within a group.

Usage

```
add_nesttrait(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: NestTrait v2 (Chia et al. 2023, Scientific Data, CC-BY 4.0).

Value

The same data.frame with three collapsed categorical columns (`nesttrait_structure`, `nesttrait_site`, `nesttrait_attachment`, each a pipe-delimited set of modalities for multi-modal species) plus 20 raw 0/1 indicator columns prefixed `nesttrait_`: `brood_parasite`, `mound_builder`, seven `nestsite_*`, seven `neststr_*` and four `nestatt_*` flags.

References

Chia SY et al. (2023) A global database of bird nest traits. Scientific Data. doi:10.1038/s41597-023028371

Examples

```
## Not run:
taxify("Turdus merula", backbone = "gbif") |>
  add_nesttrait()

## End(Not run)
```

---

add_noddb *Add root-nodule nitrogen fixation (NodDB)*

Description

Joins the nodulation status of a plant genus to a `taxify()` result. NodDB records nodulation per genus, so the join is on genus, the same grain as `add_fungalroot()`.

Usage

```
add_noddb(x, cols = NULL, verbose = TRUE)
```

Arguments

x	A data.frame returned by `taxify()`.
cols	Which columns to attach: `NULL` (default) all, or a character vector of names. See `enrichment_cols`.
verbose	Logical. Default `TRUE`.

Details

Source: Tedersoo et al. (2018), 824 plant genera. CC BY-SA 4.0.

`nodulation_consensus` is the authors' verdict verbatim, which distinguishes a confirmed record from a phylogenetic inference (`Rhizobia` against `likely_Rhizobia`) and carries their negative calls as `unlikely_Rhizobia` and `unlikely_Frankia`. `nodulation_type` collapses that to the symbiont involved (`rhizobia`, `frankia`, `nostocaceae`, `present`, `none`) reading both `unlikely_` verdicts as `none`.

`positive_reports` and `negative_reports` are the counts of published observations behind the verdict, so a thinly evidenced call is visible.

Value

The same data.frame with `noddb_` columns: `nodulation_consensus`, `nodulation_type`, `nodulation_clade`, `nodulation_family`, `spp_recognized`, `spp_studied`, `positive_reports`, `negative_reports`.

References

Tedersoo L, Laanisto L, Rahimlou S, Toussaint A, Hallikma T, Partel M (2018) Global database of plants with root-symbiotic nitrogen fixation: NodDB. Journal of Vegetation Science 29:560-568. doi:10.1111/jvs.12627

Data: doi:10.15156/BIO/587469

Examples

```
## Not run:
taxify(c("Pisum sativum", "Alnus glutinosa", "Quercus robur")) |>
  add_noddb()

## End(Not run)
```

`add_nztd` *Add NZ marine benthos traits (NZTD)*

Description

Joins New Zealand marine benthic-invertebrate functional traits to a `taxify()` result by `accepted_name`. Each fuzzy-coded trait is reduced to its dominant modality.

Usage

```
add_nztd(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: NZTD (Lam-Gordillo et al. 2023, figshare, CC-BY 4.0).

Value

The same data.frame with categorical `nztd_` columns: `bioturbation`, `body_size`, `degree_of_attachment`, `feeding_mode`, `living_habit`, `mobility`, `morphology`, `movement_method`, `rigidity`.

References

Lam-Gordillo O et al. (2023) New Zealand Trait Database (NZTD) for marine benthic invertebrates. figshare. doi:10.6084/m9.figshare.21939647

Examples

```
## Not run:
taxify("Macomona liliana", backbone = "gbif") |>
  add_nztd()

## End(Not run)
```

`add_octocoral` *Add octocoral traits (Octocoral Trait Database)*

Description

Joins soft-coral (octocoral) colony, polyp, skeleton, symbiosis and feeding traits to a `taxify()` result by `accepted_name`. Built from long-format records reduced to one value per species.

Usage

```
add_octocoral(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Octocoral Trait Database v2.2 (Gomez-Gras et al., CC-BY 4.0).

Value

The same data.frame with `octocoral_` columns: numeric `colony_height`, `colony_width`, `tentacles_per_polyp`; categorical `growth_form`, `type_of_growth`, `type_of_skeleton`, `polyp_retractability`, `polyp_dimorphism`, `zooxanthellate`, `axis_presence`, `feeding_mechanism`, `coloniality`, `skeletal_rigidity`, `calcareous_sclerites_pr`

References

Octocoral Trait Database v2.2. Zenodo. doi:10.5281/zenodo.14228404

Examples

```
## Not run:
taxify("Gorgonia ventalina", backbone = "gbif") |>
  add_octocoral()

## End(Not run)
```

add_odonata *Add odonate behavioural/ecological traits (OPD)*

Description

Joins Odonate Phenotypic Database categorical traits to a `taxify()` result by `accepted_name` (modal value per species).

Usage

```
add_odonata(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Odonate Phenotypic Database (Waller et al., Dryad, CC-BY 4.0).

Value

The same data.frame with categorical `odonata_territoriality`, `odonata_flight_mode`, `odonata_mate_guarding`, `odonata_habitat_openness`, `odonata_has_wing_pigment`.

References

Waller JT et al. The Odonate Phenotypic Database. Dryad. doi:10.5061/dryad.15pm5qc

Examples

```
## Not run:
taxify("Calopteryx splendens", backbone = "gbif") |>
  add_odonata()

## End(Not run)
```

`add_pantheria` *Add mammal life-history traits (PanTHERIA)*

Description

Joins PanTHERIA mammal life-history and ecological traits to a `taxify()` result by looking up `accepted_name`.

Usage

```
add_pantheria(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: PanTHERIA (Jones et al. 2009, Ecological Archives, CC0). Coverage: ~5.4k mammal species. Mammals only.

Value

The same data.frame with additional columns:

pantheria_body_mass_g Adult body mass in grams.

longevity_mo Maximum longevity in months.

litter_size Litter size (mean).

gestation_d Gestation length in days.

weaning_d Weaning age in days.

home_range_km2 Home range size in km^2.

diet_breadth Diet breadth (number of diet categories).

habitat_breadth Habitat breadth (number of habitat types).

References

Jones KE et al. (2009) PanTHERIA: a species-level database of life history, ecology, and geography of extant and recently extinct mammals. Ecology 90:2648.

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify("Vulpes vulpes", backbone = "gbif") |>
  add_pantheria()

options(old)
```

`add_parravicini` *Add reef-fish trophic guild (Parravicini)*

Description

Joins the consensus reef-fish trophic-guild assignment to a `taxify()` result by `accepted_name`. The guild is the modal expert classification.

Usage

```
add_parravicini(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Parravicini et al. (2020) reef-fish trophic guilds (PLoS Biology, CC-BY 4.0).

Value

The same data.frame with categorical `parravicini_trophic_guild`.

References

Parravicini V et al. (2020) Delineating reef fish trophic guilds with global gut content data synthesis and phylogeny. PLoS Biology 18:e3000702. doi:10.1371/journal.pbio.3000702

Examples

```
## Not run:
taxify("Zebrasoma scopas", backbone = "gbif") |>
  add_parravicini()

## End(Not run)
```

`add_pelagic` *Add pelagic species traits*

Description

Joins pelagic fish/cephalopod/gelatinous traits to a `taxify()` result by `accepted_name`.

Usage

```
add_pelagic(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Gleiber et al. (2024) Pelagic Species Trait Database (Borealis, CC-BY 4.0).

Value

The same data.frame with `pelagic_` columns: numeric `depth_min_m`, `depth_max_m`, `temp_min_c`, `temp_max_c`, `temp_mean_c`, `length_min_tl_cm`, `length_max_tl_cm`, `trophic_level`; categorical `vert_habitat`, `horz_habitat`, `body_shape`, `phys_defense`, `gregarious`.

References

Gleiber MR et al. (2024) A trait database for pelagic species. Scientific Data. doi:10.5683/SP3/0YFJED

Examples

```
## Not run:
taxify("Thunnus albacares", backbone = "gbif") |>
  add_pelagic()

## End(Not run)
```

add_phylacine *Add mammal traits including extinct species (PHYLACINE)*

Description

Joins PHYLACINE mammal traits to a `taxify()` result by looking up `accepted_name`. PHYLACINE covers extant plus recently and prehistorically extinct mammals; it is offered alongside `add_pantheria()` and `add_combine()`, not as a replacement.

Usage

```
add_phylacine(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: PHYLACINE v1.2 (Faurby et al. 2018, Ecology, CC0). Coverage: ~5.8k mammal species including extinct taxa. `Mass.g` is partly modelled (PHYLACINE gap-fills data-poor and extinct species); `phylacine_mass_method` / `phylacine_mass_method_class` record how each mass was derived, so a modelled mass is not mistaken for a measurement where PHYLACINE is the only source.

Value

The same data.frame with additional columns:

phylacine_mass_g Body mass (g).

phylacine_diet_plant_pct Percent of diet that is plant.

phylacine_diet_vertebrate_pct Percent of diet that is vertebrate.

phylacine_diet_invertebrate_pct Percent of diet that is invertebrate.

phylacine_terrestrial Terrestrial habit (0/1).

phylacine_marine Marine habit (0/1).

phylacine_freshwater Freshwater habit (0/1).

phylacine_aerial Aerial habit (0/1).

phylacine_island_endemicity Island endemicity class.

phylacine_iucn_status IUCN status (includes EP = extinct in prehistory, EX, EW).

phylacine_mass_method How the body mass was obtained (the PHYLACINE `Mass.Method` field, verbatim): `Reported`, `Imputed`, or an allometric estimate (e.g. "Assumed isometric based on head-body length").

phylacine_mass_method_class Coarse provenance of the body mass: `reported` (measured/compiled), `estimated` (allometric or same-size analogy), or `imputed` (phylogenetic gap-fill). Only `reported` is an observation; the others are model estimates.

References

Faurby S et al. (2018) PHYLACINE 1.2: The Phylogenetic Atlas of Mammal Macroecology. Ecology 99:2626. doi:10.1002/ecy.2443

Examples

```
## Not run:
taxify("Mammuthus primigenius", backbone = "gbif") |>
  add_phylacine()

## End(Not run)
```

`add_pignatti` *Add Italian plant traits from Pignatti (on demand, via TR8)*

Description

Fetches Italian Ellenberg-type indicator values, life form, and chorotype from Pignatti's Flora d'Italia (Pignatti, Menegoni & Pietrosanti 2005) for the species in a `taxify()` result, using the TR8 package, and joins them by `accepted_name`. TR8 ships these values bundled, so this works offline.

Usage

```
add_pignatti(x, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`verbose`	Logical. Default `TRUE`.

Details

These values originate in a copyrighted publication, so taxify does not redistribute them. This function reads the copy bundled in the suggested package TR8 (which redistributes it under TR8's GPL, with attribution); taxify ships none of it and no internet access is required. For European-calibration indicator values see `add_eive()`.

Value

The same data.frame with additional columns:

light_it, temperature_it, continentality_it, moisture_it, reaction_it, nutrients_it, salinity_it Ellenberg-type indicator values calibrated for the Italian flora (codes; `X` = indifferent, `0` = not applicable).

life_form_it Life form for the Italian flora.

chorotype_it Chorological type (distribution).

References

Pignatti S, Menegoni P, Pietrosanti S (2005) Bioindicazione attraverso le piante vascolari. Braun-Blanquetia 39. Bocci G (2015) TR8: an R package for easily retrieving plant species traits. Methods in Ecology and Evolution 6:347-350.

Examples

```
old <- options(taxify.data_dir = taxify_example_data())

## Not run:
# add_pignatti() fetches Italian trait data on demand via the TR8 package.
taxify("Abies alba") |>
  add_pignatti()

## End(Not run)

options(old)
```

`add_plantatt` *Add British and Irish plant attributes (PLANTATT)*

Description

Joins attributes of British and Irish vascular plants (Hill et al. 2004) to a `taxify()` result by looking up `accepted_name`.

Usage

```
add_plantatt(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

1,887 taxa. For the German-flora equivalent see `add_floraweb()`, for the British Ecoflora traits `add_ecoflora()`, and for European-calibration indicator values `add_eive()`.

This source states no licence (it is copyright the Biological Records Centre), so taxify ships no pre-built copy of it. The first call builds it from the original source on the user's machine, which requires the taxifydb package (`remotes::install_github("gcol33/taxifydb")`). taxify redistributes none of the data. Cite Hill et al. (2004) for the data.

Value

The same data.frame with additional columns. The default set:

plantatt_ellenberg_light, plantatt_ellenberg_moisture, plantatt_ellenberg_reaction, plantatt_ellenberg_nitrogen, pla Ellenberg indicator values calibrated for the British flora.

plantatt_max_height_cm Maximum height, cm.

plantatt_life_form Life-form code, kept verbatim (`Ch`, `Ph`, `Gn`, ...).

plantatt_woodiness Woodiness code, kept verbatim (`w` woody, `h` herbaceous, `sw` semi-woody).

plantatt_native_status Native-status code, kept verbatim (`N` native, `AN` alien naturalised, `AR` archaeophyte, ...).

References

Hill MO, Preston CD, Roy DB (2004) PLANTATT: Attributes of British and Irish Plants. Biological Records Centre, Centre for Ecology and Hydrology.

Examples

```
## Not run:
# Builds the enrichment on first use (needs taxifydb).
taxify("Bellis perennis", backbone = "gbif") |>
  add_plantatt()

## End(Not run)
```

`add_plazi_collembola_body_length`

Add Collembola body length (Plazi treatments)

Description

Joins per-species springtail body length to a `taxify()` result by looking up `accepted_name`. The value is mined from Plazi taxonomic treatments and aggregated per species: the median, with the observed range and the measurement and treatment-paper counts.

Usage

```
add_plazi_collembola_body_length(x, cols = NULL, verbose = TRUE)
```

Arguments

x	A data.frame returned by taxify().
cols	Which columns to attach: NULL (default) all, or a character vector of names. See enrichment_cols.
verbose	Logical. Default TRUE.

Details

Source: Plazi TreatmentBank (plazi.org), taxonomic treatments republished as Darwin Core Archives through GBIF, CC0 1.0. Coverage: 998 species from 1,117 measurements. On the 35 species it shares with the BETSI body-length floor the mined values track BETSI at Pearson r = 0.906 (Spearman 0.955), and it adds 963 species BETSI does not cover. Cite the individual treatments for any species-level use. Body length is also available through add_trait("body_length").

Value

The same data.frame with additional columns:

plazi_body_length_mm Body length (mm), median of the mined measurements.

plazi_body_length_min_mm Smallest measurement (mm).

plazi_body_length_max_mm Largest measurement (mm).

plazi_body_length_n Number of measurements aggregated.

plazi_body_length_sources Number of distinct treatment papers.

References

Plazi TreatmentBank. https://plazi.org/

Examples

```
## Not run:
taxify("Isotoma viridis", backbone = "gbif") |>
  add_plazi_collembola_body_length()

## End(Not run)
```

`add_pottier` *Add amphibian heat tolerance (Pottier)*

Description

Joins amphibian upper thermal-limit and body-size summaries to a `taxify()` result by `accepted_name`. Per-measurement records are reduced to species medians; heat tolerance pools across metrics and acclimation conditions, so it is an approximate species-level upper thermal limit.

Usage

```
add_pottier(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Pottier et al. (2022) amphibian heat tolerance database (Scientific Data, CC-BY 4.0).

Value

The same data.frame with numeric columns `pottier_heat_tolerance_c`, `pottier_acclimation_temp_c`, `pottier_svl_mm`, `pottier_body_mass_g`.

References

Pottier P et al. (2022) A comprehensive database of amphibian heat tolerance. Scientific Data 9:600. doi:10.1038/s41597022017049

Examples

```
## Not run:
taxify("Rana temporaria", backbone = "gbif") |>
  add_pottier()

## End(Not run)
```

add_quimbayo *Add reef-fish traits (Quimbayo)*

Description

Joins Atlantic and Eastern-Pacific reef-fish life-history, ecology and behaviour traits to a `taxify()` result by `accepted_name`.

Usage

```
add_quimbayo(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Quimbayo et al. (2021) reef-fish trait database (ESA data paper; Zenodo, open).

Value

The same data.frame with `quimbayo_` columns: numeric `body_size_max_cm`, `aspect_ratio`, `trophic_level`, `depth_min_m`, `depth_max_m`, `temp_occurrence_mean_c`; categorical `home_range`, `diel_activity`, `water_level`, `body_shape`, `mouth_position`, `diet`, `spawning`, `size_group`.

References

Quimbayo JP et al. (2021) Life-history traits, geographical range, and conservation aspects of reef fishes. Ecology. doi:10.5281/zenodo.4455016

Examples

```
## Not run:
taxify("Thalassoma bifasciatum", backbone = "gbif") |>
  add_quimbayo()

## End(Not run)
```

add_ramond *Add marine protist functional traits (Ramond et al.)*

Description

Joins genus-level morphological, behavioural and ecological traits of marine protists to a `taxify()` result by looking up `genus`, so any species in a covered genus is annotated.

Usage

```
add_ramond(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Ramond et al. (SEANOE, CC BY 4.0), functional traits of marine protists. Traits are genus-level; numeric cell size is aggregated by median, categorical traits by mode.

Value

The same data.frame with additional columns. The curated set:

ramond_shape Cell shape (round, elongated, amoeboid, ...).

ramond_motility Motility mode (swimmer, floater, gliding, ...).

ramond_ingestion Ingestion / trophic mode (phagotrophic, ...).

ramond_chloroplast Chloroplast presence (1 = present).

ramond_symbiontic Symbiotic relationship (parasite, mutualist, ...).

ramond_colony Colony form.

ramond_salinity Salinity preference.

ramond_size_min_um Minimum cell size (micrometres).

ramond_size_max_um Maximum cell size (micrometres).

With `cols = "all"` the full set of behavioural, ecological and symbiosis descriptors is attached under their source names.

References

Ramond P, Siano R, Sourisseau M (2018) Functional traits of marine protists. SEANOE. doi:10.17882/51662

Examples

```
## Not run:
taxify("Alexandrium", backbone = "gbif") |>
  add_ramond()

## End(Not run)
```

`add_repttraits` *Add reptile ecological traits and distribution (ReptTraits)*

Description

Joins species-level reptile traits from ReptTraits (Oskyrko et al. 2024) to a `taxify()` result by looking up `accepted_name`. ReptTraits is built on the Reptile Database taxonomy, so it joins cleanly against the `reptiledb` backbone (and any backbone that resolves to Reptile Database accepted names).

Usage

```
add_repttraits(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

The dataset carries a per-species distribution signal (biogeographic realm, elevation range and mean climate) alongside body-size and life-history traits, across all reptiles (snakes, lizards, amphisbaenians, turtles, crocodiles and the tuatara), not lizards only.

Source: ReptTraits v1.2 (Oskyrko et al. 2024, Scientific Data, CC BY 4.0). Coverage: 12,060 reptile species. The biogeographic realm and climate fields give a coarse, realm-level range signal; they are not a fine-grained (TDWG-level) range like the plant ranges used by the `region` constraint.

Value

The same data.frame with additional columns:

biogeographic_realm Main biogeographic realm (e.g. Neotropic, Palearctic, Afrotropic, Australo-Pacific, Marine).

microhabitat Microhabitat (e.g. Terrestrial, Saxicolous, Arboreal).

habitat_type Habitat type(s) (e.g. Forest, Desert, Wetlands).

elevation_min_m Minimum recorded elevation in metres.

elevation_max_m Maximum recorded elevation in metres.

mean_annual_temp_c Mean annual temperature across the range (degrees Celsius).

insular_endemic Whether the species is insular/endemic (`"Yes"`/`"No"`).

body_mass_g Maximum body mass in grams.

svl_mm Maximum snout-vent length (straight carapace length for turtles) in mm.

total_length_mm Maximum total length in mm.

longevity_yr Maximum longevity in years.

diet Diet category (e.g. Carnivorous, Herbivorous, Omnivorous).

reproductive_mode Reproductive mode (oviparous/viviparous/...).

clutch_size Mean clutch or litter size.

active_time Activity time (Diurnal/Nocturnal/Cathemeral).

foraging_mode Foraging mode (ACT active / AMB ambush / Mixed).

References

Oskyrko O, Mi C, Meiri S, Du W (2024) ReptTraits: a comprehensive dataset of ecological traits in reptiles. Scientific Data 11:243. doi:10.1038/s41597024030795

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify("Pogona vitticeps", backbone = "reptiledb") |>
  add_repttraits()

options(old)
```

---

add_rimet_phyto *Add phytoplankton cell metrics (Rimet & Druart)*

Description

Joins cell-level morphometrics for temperate-lake phytoplankton to a `taxify()` result by `accepted_name`.

Usage

```
add_rimet_phyto(x, cols = NULL, verbose = TRUE)
```

Arguments

x	A data.frame returned by taxify().
cols	Which columns to attach: NULL (default) the curated set, "all" every column the source carries, or a character vector of names. See enrichment_cols.
verbose	Logical. Default TRUE.

Details

Source: Rimet & Druart (2018) phytoplankton metrics database (Zenodo, CC-BY 4.0).

Value

The same data.frame with numeric columns rimet_phyto_cell_length_um, rimet_phyto_cell_width_um, rimet_phyto_cell_thickness_um, rimet_phyto_cell_surface_area_um2, rimet_phyto_cell_biovolume_um3.

References

Rimet F, Druart JC (2018) A trait database for phytoplankton of temperate lakes. Zenodo. doi:10.5281/zenodo.1164834

Examples

```
## Not run:
taxify("Asterionella formosa", backbone = "gbif") |>
  add_rimet_phyto()

## End(Not run)
```

---

add_saproxylic *Add saproxylic beetle morphology (Hagge)*

Description

Joins European deadwood-beetle body and appendage morphometrics to a taxify() result by accepted_name.

Usage

```
add_saproxylic(x, cols = NULL, verbose = TRUE)
```

Arguments

x	A data.frame returned by taxify().
cols	Which columns to attach: NULL (default) the curated set, "all" every column the source carries, or a character vector of names. See enrichment_cols.
verbose	Logical. Default TRUE.

Details

Source: Hagge et al. (2021) saproxylic beetle morphology (Dryad, CC0).

Value

The same data.frame with numeric `saproxylic_` columns: `body_length_mm`, `body_width_mm`, `body_height_mm`, `mass_mg`, `colour_lightness`, `head_length_mm`, `pronotum_length_mm`, `elytra_length_mm`, `wing_length_mm`, `wing_aspect`, `antenna_length_mm`, `eye_length_mm`.

References

Hagge J et al. (2021) Morphological trait database of European saproxylic beetles. Dryad. doi:10.5061/dryad.2fqz612p3

Examples

```
## Not run:
taxify("Rhysodes sulcatus", backbone = "gbif") |>
  add_saproxylic()

## End(Not run)
```

`add_sealifebase` *Add aquatic-life traits (SeaLifeBase)*

Description

Joins SeaLifeBase morphological and ecological traits to a `taxify()` result by looking up `accepted_name`. SeaLifeBase is the non-fish companion to FishBase: molluscs, crustaceans, echinoderms, marine mammals, reptiles and other aquatic organisms. For fishes, use `add_fishbase()`.

Usage

```
add_sealifebase(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: SeaLifeBase via rfishbase (Palomares & Pauly, CC BY-NC 4.0). Non-fish aquatic life only.

The `sb_lw_*` columns give one representative length-weight fit per species from the POPLW table, so a length can be converted to a mass where `sb_body_mass_g` (a record maximum) does not fit the analysis.

The build-from-source fallback requires the **rfishbase** package (available on CRAN). Pre-built `.vtr` files do not require rfishbase.

Value

The same data.frame with additional columns:

sb_body_length_cm Maximum body length in centimetres.

sb_body_mass_g Maximum published weight in grams (SeaLifeBase `SPECIES.Weight`). This is a record maximum, not a typical or adult-mean mass.

sb_trophic_level Trophic level.

sb_depth_min_m Minimum depth in metres.

sb_depth_max_m Maximum depth in metres.

sb_vulnerability Vulnerability index (0–100).

sb_habitat Habitat type (e.g. benthic, pelagic).

sb_importance Commercial importance category.

sb_lw_a Coefficient `a` of the length-weight relationship `W = a * L^b` (weight in g, length in cm of type `sb_lw_type`), from SeaLifeBase's POPLW table.

sb_lw_b Exponent `b` of the length-weight relationship.

sb_lw_type Length convention the coefficients were fitted against (`TL` total, `SL` standard, `WD` width, ...). Chosen to match the species' maximum-length type where recorded, so it applies to `sb_body_length_cm`. Applying a coefficient to a different length type is a silent error.

sb_lw_method How the fit was obtained (e.g. "type I linear regression", "single L-W pair with b=3").

sb_lw_sex Sex the fit applies to (unsexed, mixed, female, male, juvenile).

sb_lw_n Sample size the fit was based on.

sb_lw_r2 Coefficient of determination (`r^2`) of the fit.

References

Palomares MLD, Pauly D (eds.) (2024) SeaLifeBase. World Wide Web electronic publication, https://www.sealifebase.org.

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify("Octopus vulgaris", backbone = "gbif") |>
```

```
  add_sealifebase()

options(old)
```

`add_sharkipedia` *Add elasmobranch life-history traits (Sharkipedia)*

Description

Joins Sharkipedia shark and ray life-history traits to a `taxify()` result by `accepted_name`. Long-format observations are reduced to one value per species (numeric traits by median) at build time.

Usage

```
add_sharkipedia(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Sharkipedia (Mull et al. 2022, Scientific Data, CC-BY 4.0).

Value

The same data.frame with additional columns:

sharkipedia_lmax_cm Maximum observed length (cm).

sharkipedia_vbgf_linf_cm von Bertalanffy asymptotic length Linf (cm).

sharkipedia_vbgf_k von Bertalanffy growth coefficient k (per year).

sharkipedia_vbgf_t0 von Bertalanffy t0 (years).

sharkipedia_length_first_maturity_cm Length at first maturity (cm).

sharkipedia_length_birth_cm Length at birth (cm).

sharkipedia_amax_observed_yr Maximum observed age (years).

sharkipedia_age_first_maturity_yr Age at first maturity (years).

sharkipedia_uterine_fecundity Uterine fecundity.

sharkipedia_gestation_length Gestation length.

sharkipedia_natural_mortality Natural mortality M.

References

Mull CG et al. (2022) Sharkipedia: a curated open access database of shark and ray life history traits and abundance time-series. Scientific Data 9:559. doi:10.1038/s41597022016551

Examples

```
## Not run:
taxify("Carcharodon carcharias", backbone = "gbif") |>
  add_sharkipedia()

## End(Not run)
```

`add_sheld` *Add freshwater mussel traits (SHELD)*

Description

Joins US freshwater-mussel life-history and host traits to a `taxify()` result by `accepted_name`.

Usage

```
add_sheld(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: SHELD (Hopper et al. 2023, Scientific Data, CC-BY 4.0).

Value

The same data.frame with `sheld_` columns: numeric `mean_length_mm`, `max_length_mm`, `mature_age`, `max_age`, `growth_rate`, `fecundity`, `n_host_species`, `n_host_family`; categorical `brood`, `marsupial_gills`, `hermaphrodite`, `shell_sculpture`.

References

Hopper GW et al. (2023) A trait dataset for freshwater mussels of the United States of America. Scientific Data 10:745. doi:10.1038/s41597023026359

Examples

```
## Not run:
taxify("Lampsilis cardium", backbone = "gbif") |>
  add_sheld()

## End(Not run)
```

`add_spider_traits` *Add spider traits (World Spider Trait Database)*

Description

Joins species-level spider morphometric and ecological traits to a `taxify()` result by looking up `accepted_name`. Values are aggregated from the World Spider Trait Database (numeric traits by median, categorical traits by mode); access-restricted source records are excluded.

Usage

```
add_spider_traits(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: World Spider Trait Database (Pekar et al. 2021, Database, CC BY 4.0). Coverage: ~7.3k spider species. Morphometry is sexually dimorphic in spiders; the value here is the across-record median and is not split by sex.

Value

The same data.frame with additional columns:

spider_body_length_mm Body length (mm).

spider_prosoma_length_mm Cephalothorax (prosoma) length (mm).

spider_prosoma_width_mm Cephalothorax (prosoma) width (mm).

spider_abdomen_length_mm Abdomen (opisthosoma) length (mm).

spider_leg1_length_mm Leg I length (mm).

spider_ballooning Ballooning (aerial dispersal): yes/no.

spider_web_building Web building: yes/no.

spider_hunting_guild Hunting guild.

spider_web_type Web type.

spider_circadian_activity Circadian activity (diurnal/nocturnal).

spider_stratum Vertical stratum (habitat layer).

References

Pekar S et al. (2021) The World Spider Trait database: a centralized global open repository for curated data on spider traits. Database 2021:baab064. doi:10.1093/database/baab064

Examples

```
## Not run:
taxify("Araneus diadematus", backbone = "gbif") |>
  add_spider_traits()

## End(Not run)
```

`add_sworm` *Add earthworm ecological groups (sWorm)*

Description

Joins the Bouche ecological group of an earthworm species to a `taxify()` result by accepted name.

Usage

```
add_sworm(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) all, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Phillips et al. (2021), 171 species. CC BY 4.0. Earthworms are otherwise absent from the bundled trait sources.

The release is a community dataset covering 10,840 sites, and its one species-level trait is the ecological group each earthworm was assigned from feeding and burrowing behaviour. That assignment is constant within a species across the whole compilation, so collapsing it to one row per species loses nothing.

Abundance and wet biomass are not carried: both ship several units in one column (individuals, individuals per m2 and per m3; grams, g/m2 and mg/m2), mixing a per-individual measure with a per-area density, so neither survives aggregation to the species. The native/non-native flag is a status at a site rather than a property of the species; `add_griis()` covers introduction status per country.

Value

The same data.frame with `sworm_ecological_group`: one of `epigeic`, `endogeic`, `anecic`, `epi-endogeic`.

References

Phillips HRP, Guerra CA, Bartz MLC, et al. (2021) Global data on earthworm abundance, biomass, diversity and corresponding environmental properties. Scientific Data 8:136. doi:10.1038/s41597-02100912z

Examples

```
## Not run:
taxify(c("Lumbricus terrestris", "Aporrectodea caliginosa")) |>
  add_sworm()

## End(Not run)
```

`add_tetradensity` *Add population density (TetraDENSITY)*

Description

Joins species-median terrestrial-vertebrate population density to a `taxify()` result by `accepted_name`. Only `ind/km2` records are used.

Usage

```
add_tetradensity(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Santini et al. TetraDENSITY (figshare, CC-BY 4.0). Records in other density units are excluded to avoid mixing.

Value

The same data.frame with numeric `tetradensity_density_ind_km2`.

References

Santini L et al. TetraDENSITY: a database of population density estimates in terrestrial vertebrates. figshare. doi:10.6084/m9.figshare.5371633

Examples

```
## Not run:
taxify("Capreolus capreolus", backbone = "gbif") |>
  add_tetradensity()

## End(Not run)
```

`add_thermofresh` *Add freshwater thermal-tolerance traits (ThermoFresh)*

Description

Joins species-level critical thermal limits for freshwater fish, invertebrates and amphibians to a `taxify()` result by looking up `accepted_name`. All values are in degrees Celsius.

Usage

```
add_thermofresh(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: the Freshwater thermal-tolerance database (Helena Bayat and contributors, Zenodo, CC BY 4.0). Each source record is one tolerance test; values are reduced to species-level medians per metric.

Value

The same data.frame with additional columns:

thermofresh_ctmax Critical thermal maximum (degrees C).

thermofresh_ctmin Critical thermal minimum (degrees C).

thermofresh_lt50 Median lethal temperature (degrees C).

thermofresh_ltmax Lethal thermal maximum (degrees C).

thermofresh_ltmin Lethal thermal minimum (degrees C).

References

Freshwater thermal-tolerance database. Zenodo. doi:10.5281/zenodo.14056760

Examples

```
## Not run:
taxify("Salmo trutta", backbone = "gbif") |>
  add_thermofresh()

## End(Not run)
```

`add_trait` *Add a trait from every source that carries it*

Description

Attaches a single harmonized trait (e.g. woodiness, plant height) to a `taxify()` result, pulling from every enrichment source that provides it and reconciling their differing vocabularies and units. Each `add_<source>()` function joins one dataset; `add_trait()` takes the name of a trait and gathers it from every source that carries it.

Usage

```
add_trait(
  x,
  trait,
  sources = "all",
  mode = c("coalesce", "wide"),
  combine = NULL,
  priority = NULL,
  provenance = FALSE,
  verbose = TRUE,
  aggregate_trait_fallback = getOption("taxify.aggregate_trait_fallback", TRUE)
)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`trait`	Character. A single trait name; see `list_traits()` for the available traits and `trait_info()` for a trait's sources and units.
`sources`	Which sources to use. Either the string `"all"` (the default) for every source registered for the trait, or a character vector of source names (see `trait_info()`).
`mode`	One of `"coalesce"` (default) or `"wide"`. `"coalesce"` reduces the sources to one value per row (see `combine`). `"wide"` attaches one harmonized column per source.

combine
: How mode = "coalesce" reduces the per-source values for a row. NULL (default) is method-aware: when the trait's sources agree in method it uses "median" for numeric traits and "first" for categorical traits; when sources measure the trait differently (a source carries a caution, e.g. maximum vs fine-root diameter) it uses "complete" instead of blending them. Numeric options: "median", "mean", "first" (highest-priority source that has a value), "min", "max", "complete" (the single most populated source, reported verbatim). Categorical options: "first", "vote" (majority across sources, ties broken by priority), or "complete". Median is the numeric default for concordant sources because trait values are skewed, so a single outlier should not decide the value; "complete" is the default for discordant sources because a median across methods matches no method. Passing combine explicitly overrides this and is applied to all sources.

priority
: Character vector of source names giving the priority order (highest priority first), used by combine = "first", for tie-breaking combine = "vote", and to break ties in combine = "complete". Only used when mode = "coalesce"; defaults to the registered order for the trait (see trait_info()).

provenance
: Logical. When TRUE, also return the references behind each value, for sources whose build records them (AusTraits, GIFT, BROT and LEDA): <trait>_refs in mode = "coalesce", holding the references of the sources that produced the reported value, and <trait>_<source>_refs in mode = "wide". Ids are <enrichment>:<id>, several joined by |; pass the column to cite() to resolve them to citations. A source that records no per-value references contributes none. Default FALSE.

verbose
: Logical. Default TRUE.

aggregate_trait_fallback
: Logical. When an aggregate or hybrid name has no trait record of its own, fall back to the underlying binomial. Defaults to getOption("taxify.aggregate_trait_fallback", TRUE), so it can be set per call or for the session. Grain-pinned sources are unaffected.

Details

Each source keeps its provenance. The default "coalesce" mode reduces the sources to one value per row plus the columns that document it: its unit, the sources that contributed, how many, and a caution when the sources measure the trait by different methods. The opt-in "wide" mode instead gives every source its own column (<trait>_<source>), so per-source agreement and conflict stay fully visible.

Harmonization is per source: a categorical source is mapped to the trait's shared vocabulary, and a numeric source is converted to the trait's canonical unit. For example, GIFT seed mass (grams) and Diaz et al. seed mass (milligrams) both arrive as milligrams. The mappings and units for a trait are listed by trait_info().

Some sources report the same trait in the right unit but under a different definition (for example AusTraits root diameter is a maximum including coarse roots, while GRooT is fine-root only). Such a source carries a caution in the registry. In mode = "coalesce" with the default combine, a trait whose sources disagree in method is not blended: the most complete source is reported and <trait>_caution records the difference. trait_info() lists each source's harmonization note and caution.

With provenance = TRUE, a reference names the record a database took the value from: for a categorical value the references that state that value, for a numeric value every reference entering the source's aggregate. Under combine = "median" or "mean" the coalesced references are those of every contributing source; under "first" or "complete" those of the one source reported; under "vote", "min" or "max" those of the sources whose value is the one reported.

A source enrichment that is not installed and cannot be downloaded or built is skipped with a warning, and the trait is assembled from the sources that are available.

Value

The same data.frame with added columns.

mode = "coalesce" <trait> (the reduced value); <trait>_unit (the canonical unit, numeric traits only); <trait>_sources (the source it came from, or the comma-separated contributing sources with an aggregating combine); <trait>_n (how many sources backed the value); for numeric traits, <trait>_min and <trait>_max (the range of values behind the headline, across the contributing sources, and across a source's own records where that spread was recorded at build time, so a life-stage or population span stays visible); and, only when a source measured the trait differently, <trait>_caution explaining the method difference; with provenance = TRUE, <trait>_refs. To inspect every source, use mode = "wide".

mode = "wide" One column per source, <trait>_<source>, each harmonized to the trait's shared vocabulary (categorical) or unit (numeric); <trait>_unit; <trait>_caution on rows where a cautioned source supplied a value; and, with provenance = TRUE, <trait>_<source>_refs per source.

Numeric traits are returned in the trait's canonical unit (see trait_info()); rows absent from a source get NA.

See Also

list_traits() to see available traits, trait_info() for a trait's sources and units. The per-source functions (add_zanne(), add_gift(), add_diaz_traits(), add_leda()) join one dataset each.

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

# One coalesced value plus its provenance (unit, sources, count):
taxify("Abies alba") |>
  add_trait("seed_mass")

# One column per source, to inspect agreement and conflict:
taxify("Abies alba") |>
  add_trait("woodiness", mode = "wide")

# The references behind each value, resolved to citations:
res <- taxify("Abies alba") |>
  add_trait("seed_mass", mode = "wide", provenance = TRUE)
cite(res$seed_mass_gift_refs)
```

```
options(old)
```

`add_tree_of_sex` *Add sex-determination traits (Tree of Sex)*

Description

Joins sexual-system and sex-determination traits to a `taxify()` result by looking up `accepted_name`. Covers plants, vertebrates and invertebrates; some traits are group-specific (selfing for plants, environmental sex determination for vertebrates, haplodiploidy for invertebrates).

Usage

```
add_tree_of_sex(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Tree of Sex (Tree of Sex Consortium 2014, Scientific Data, CC0). Coverage: ~37.5k species across plants, vertebrates and invertebrates.

Value

The same data.frame with additional columns:

tos_taxon_group Source group (plants/vertebrates/invertebrates).

tos_sexual_system Sexual system (vocabulary differs by group).

tos_karyotype Sex-chromosome system (XY/ZW/XO/homomorphic/...).

tos_genotypic Heterogamety (male/female heterogametic/GSD/...).

tos_molecular_basis Molecular basis (Y dominant/W dominant/dosage).

tos_selfing Selfing (plants; self compatible/incompatible).

tos_environmental_sd Environmental sex determination (vertebrates; TSD/...).

tos_haplodiploidy Haplodiploidy (invertebrates).

References

The Tree of Sex Consortium (2014) Tree of Sex: a database of sexual systems. Scientific Data 1:140015. doi:10.1038/sdata.2014.15

Examples

```
## Not run:
taxify("Silene latifolia", backbone = "gbif") |>
  add_tree_of_sex()

## End(Not run)
```

`add_usda_fungus_host` *Add fungal host breadth (USDA Fungus-Host Dataset)*

Description

Joins per-fungus host breadth from the USDA National Fungus Collections Fungus-Host Dataset to a `taxify()` result by looking up `accepted_name`. Each fungus species is summarised by how many distinct host plants and host plant genera it has been recorded on.

Usage

```
add_usda_fungus_host(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Farr, Rossman & Castlebury (2021) United States National Fungus Collections Fungus-Host Dataset, Ag Data Commons (U.S. Public Domain). Fungi only; ~99k species with at least one recorded host.

Value

The same data.frame with additional columns:

fungus_host_count Number of distinct host plant species recorded.

fungus_host_genus_count Number of distinct host plant genera recorded.

References

Farr DF, Rossman AY, Castlebury LA (2021) United States National Fungus Collections Fungus-Host Dataset. Ag Data Commons. doi:10.15482/USDA.ADC/1524414

Examples

```
## Not run:
taxify("Puccinia graminis", backbone = "gbif") |>
  add_usda_fungus_host()

## End(Not run)
```

`add_useful_plants` *Add human-use categories (World Checklist of Useful Plant Species)*

Description

Joins plant human-use categories to a `taxify()` result by looking up `accepted_name`. Each of the ten Level-1 use categories is a 0/1 flag, plus a crop-wild-relative flag.

Usage

```
add_useful_plants(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Character vector of use-category columns to attach, or `"all"`. `NULL` (default) attaches all eleven. See `enrichment_cols()`.
`verbose`	Logical. Default `TRUE`.

Details

Source: World Checklist of Useful Plant Species (Diazgranados et al. 2020, KNB, CC BY 4.0). Coverage: ~39k plant species.

Value

The same data.frame with additional columns:

useful_animal_food Animal food (0/1).

useful_environmental_uses Environmental uses (0/1).

useful_fuels Fuels (0/1).

useful_gene_sources Gene sources (0/1).

useful_human_food Human food (0/1).

useful_invertebrate_food Invertebrate food (0/1).

useful_materials Materials (0/1).

useful_medicines Medicines (0/1).

useful_poisons Poisons (0/1).

useful_social_uses Social uses (0/1).

useful_crop_wild_relative Crop wild relative (0/1).

References

Diazgranados M et al. (2020) World Checklist of Useful Plant Species. Knowledge Network for Biocomplexity. doi:10.5063/F1CV4G34

Examples

```
## Not run:
taxify("Acorus calamus", backbone = "gbif") |>
  add_useful_plants()

## End(Not run)
```

`add_virion` *Add host-virus association breadth (VIRION)*

Description

Joins per-host counts of recorded virus associations to a `taxify()` result.

Usage

```
add_virion(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) all, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Carlson et al. (2022), 4223 vertebrate hosts and 9833 viruses. ODbL-1.0. As with `add_globi()` and `add_invacost()`, only the derived per-host counts are redistributed, not the association records.

These counts measure how much a host has been looked at as much as what infects it. *Homo sapiens* leads with 936 distinct viruses across 633,053 records, and the ordering below it tracks livestock and laboratory species. `virus_record_count` travels alongside so the effort behind a richness is visible; treat richness as association breadth as recorded, not as a biological property of the host.

Value

The same data.frame with `virion_` columns: `virus_richness`, `virus_family_count`, `virus_record_count`, `host_class`.

References

Carlson CJ, Gibb RJ, Albery GF, et al. (2022) The Global Virome in One Network (VIRION): an atlas of vertebrate-virus associations. mBio 13:e0298521. doi:10.1128/mbio.0298521

Examples

```
## Not run:
taxify(c("Sus scrofa", "Myotis lucifugus")) |>
  add_virion()

## End(Not run)
```

---

add_wcvp *Add WCVP native range status*

Description

Joins WCVP (World Checklist of Vascular Plants, Kew) native range data to a `taxify()` result, filtered by TDWG botanical region.

Usage

```
add_wcvp(x, region, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`region`	Character. TDWG Level 3 region code(s), or `"all"`. See `taxify_regions()` for the full list of codes. • Single code (e.g., `"BGM"` for Belgium): adds `native_status` column (no suffix). • Multiple codes (e.g., `c("BGM", "GER")`): adds `native_status_BGM`, `native_status_GER`. • `"all"`: adds one column per region in the dataset. List the region codes WCVP covers with `enrichment_groups("wcvp")`.
`cols`	Which columns to attach. `NULL` (the default) attaches the curated set; a character vector of column names attaches just those, and `"all"` attaches every column the source carries (see `enrichment_cols()`).
`verbose`	Logical. Default `TRUE`.

Details

Source: WCVP (Kew, CC BY). Coverage: ~340k plant species. Plants only.

The range reported is that of the taxon `taxify()` matched, as its backbone circumscribes it: the matched name's own records plus those of every name the backbone sinks into it. Where these disagree on a region, `native` wins over `introduced`, and `introduced` over `extinct`, since the taxon is native wherever any part of it is.

A region is left `NA` rather than filled from a broader taxon, or from one the matched backbone accepts as separate, when that is the only name another backbone offers for it. Those names are listed as `refused` in the `wcvp` entry of `attr(x, "taxify_meta")$enrichments` (the queried and refused name, the reason, the backbones offering it and the number of regions it would have filled), and counted by `base::summary()`.

Value

The same data.frame with additional column(s):

native_status One of `"native"`, `"introduced"`, `"extinct"`, or `NA` if not recorded for that region.

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify("Quercus robur") |>
  add_wcvp(region = "GER")

taxify("Quercus robur") |>
  add_wcvp(region = c("GER", "BGM"))

options(old)
```

`add_wfo_info` *Add WFO-specific columns*

Description

Joins extra World Flora Online columns to a `taxify()` result by looking up `taxon_id` in the WFO backbone.

Usage

```
add_wfo_info(x)
```

Arguments

`x`	A data.frame returned by `taxify()` with backbone == `"wfo"`.

Value

The same data.frame with additional columns:

scientificNameID WFO scientificNameID.

parentNameUsageID WFO parentNameUsageID.

name_published_in Publication reference, under the name every backbone's `.vtr` and `add_gbif_info()` use.

higherClassification Higher classification string.

taxonRemarks Taxonomic remarks.

infraspecificEpithet Infraspecific epithet (for subspecies, varieties, forms).

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify("Quercus robur") |>
  add_wfo_info()

options(old)
```

---

add_zanne *Add woodiness (Zanne et al. 2014)*

Description

Joins the woody / herbaceous classification of Zanne et al. (2014) to a `taxify()` result by looking up `accepted_name`. This function is named after its source, the Zanne Global Woodiness Database; for woodiness reconciled across every source that carries it (Zanne, GIFT), use `add_trait()` with `"woodiness"`.

Usage

```
add_zanne(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Zanne et al. 2014, Nature (Dryad, CC0). Coverage: ~50k plant species. Plants only.

Value

The same data.frame with an additional column:

woodiness One of `"woody"`, `"herbaceous"`, `"variable"`, or `NA` if not in the dataset.

References

Zanne AE et al. (2014) Three keys to the radiation of angiosperms into freezing environments. Nature 506:89-92.

See Also

`add_trait()` for woodiness harmonized across sources.

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify("Quercus robur") |>
  add_zanne()

options(old)
```

`add_zooplankton` *Add marine zooplankton traits*

Description

Joins global marine-zooplankton traits to a `taxify()` result by `accepted_name` (species-level summaries).

Usage

```
add_zooplankton(x, cols = NULL, verbose = TRUE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`cols`	Which columns to attach: `NULL` (default) the curated set, `"all"` every column the source carries, or a character vector of names. See `enrichment_cols`.
`verbose`	Logical. Default `TRUE`.

Details

Source: Pata & Hunt global marine zooplankton trait database (Zenodo, CC-BY-SA 4.0).

Value

The same data.frame with `zooplankton_` columns: numeric `body_length_max_mm`, `carbon_weight_mg`, `nitrogen_pdw_pct`; categorical `vertical_distribution`, `reproduction_mode`, `trophic_group`, `feeding_mode`, `myelination`, `habitat_association`, `diel_vertical_migration`, `bioluminescence`.

References

Pata PR, Hunt BPV (2025) A global trait database for marine zooplankton. Zenodo. doi:10.5281/zenodo.8102913

Examples

```
## Not run:
taxify("Calanus finmarchicus", backbone = "gbif") |>
  add_zooplankton()

## End(Not run)
```

`children` *List the accepted taxa within a genus or family*

Description

Returns the accepted taxa a backbone places inside a genus or family, so a checklist can be built from the backbone as well as checked against it. The parent's rank is read from the backbone, as in `downstream()`, restricted to genus and family: `children()` is `downstream()` for a genus or family parent.

Usage

```
children(
  taxon,
  backbone = NULL,
  rank = "species",
  kingdom = NULL,
  verbose = TRUE
)
```

Arguments

`taxon`	A single genus or family name.
`backbone`	A single backbone name (e.g. `"wfo"`) or a `taxify_backend` object. `NULL` (default) uses the highest-priority installed backbone.
`rank`	Rank of the children to return (`"species"` by default), or `"any"` (or `NULL`) for every rank below the parent. The parent itself is never returned.

kingdom	Optional kingdom (or kingdoms) the parent belongs to, as in taxify() ("plantae", "animals", ...). A genus or family name can be used in more than one kingdom (*Morus* is both mulberries and gannets); kingdom picks one. Without it such a result mixes the kingdoms, with a warning.
verbose	Logical. Default TRUE.

Value

A data.frame of accepted taxa, columns: name, authorship, rank, kingdom_group, family, genus, taxon_id, parent, parent_rank ("genus" or "family"), backbone. Empty if the parent is not found.

See Also

downstream() to reach below a higher rank, synonyms(), taxify().

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

children("Quercus", backbone = "wfo")

options(old)
```

---

cite *Cite data sources used in a taxify result*

Description

Prints formatted citations for the taxonomic backbone(s), enrichment datasets, and the taxify package itself. Optionally writes a BibTeX file.

Usage

```
cite(x, ...)

## Default S3 method:
cite(x, file = NULL, ...)

## S3 method for class 'character'
cite(x, file = NULL, source = NULL, ...)
```

Arguments

x	A taxify_result object, or a character vector of reference ids (cells of <enrichment>:<id> joined by |, as in <trait>_refs).
...	Unused.
file	Optional file path. If provided, BibTeX entries are written to this file (extension should be .bib).
source	For a character x only: NULL (default) when the ids carry their enrichment prefix, or the enrichment the bare ids belong to (e.g. "austraits" for the dispersal_syndrome_source column of add_austraits(cols = "all")).

Details

Given reference ids instead of a result, cite() resolves them to the works a trait value was taken from: the per-value references add_trait() returns with provenance = TRUE (<trait>_refs), or the <col>_source column of an add_<source>() function read with source =.

Value

For a taxify_result, x, invisibly (pipe-friendly). For reference ids, invisibly, a data.frame with one row per distinct id: ref (the qualified id), source, ref_id, citation, doi, and any further columns the source's reference table carries.

Examples

```
old <- options(taxify.data_dir = taxify_example_data())

result <- taxify("Quercus robur", backbone = "wfo")
result |> cite()
result |> cite(file = tempfile(fileext = ".bib"))

options(old)
```

---

class2tree *Build a taxonomy tree from resolved names*

Description

Resolves the names, attaches their full higher classification, and assembles a taxonomy tree (kingdom -> phylum -> class -> order -> family -> genus -> species) from the shared lineages. Returns the tree as a Newick string and the underlying classification table; when the **ape** package is installed, an ape phylo object is included too.

Usage

```
class2tree(x, backbone = NULL, ..., verbose = TRUE)

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

Arguments

x	Character vector of names, or a taxify() result.
backbone	Backend passed to taxify() when x is raw names. NULL (default) uses every installed backbone. Ignored when x is a result.
...	Ignored, present for S3 print method consistency.
verbose	Logical. Default TRUE.

Value

An object of class taxify_tree: a list with

newick The Newick string (internal nodes labelled by rank value, tips by species name). Characters Newick reserves, the space among them, are written as _ (Quercus_robur).

classification The classification data.frame the tree was built from.

tip_labels The species at the tips, as named in classification.

phylo An **ape** phylo object, or NULL if **ape** is not installed / the string could not be parsed. Its tip.label and node.label carry the names themselves, as tip_labels does.

See Also

lowest_common(), add_classification(), taxify().

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

tr <- class2tree(c("Quercus robur", "Quercus petraea", "Quercus pyrenaica"))
tr$newick

options(old)
```

---

comm2sci *Resolve common (vernacular) names to scientific names*

Description

The reverse of add_common_names(): given a common name, return the accepted scientific name(s) it refers to. Reads the bundled common_names enrichment (GBIF, NCBI, and Open Tree vernaculars) offline; the first call may trigger the one-time download. A common name is frequently ambiguous (several species share "bluebell", "robin"), so the result can carry more than one row per query.

Usage

```
comm2sci(
  x,
  lang = NULL,
  output = c("lookup", "result"),
  backbone = NULL,
  ...,
  verbose = TRUE
)
```

Arguments

x	Character vector of common names.
lang	Character. Restrict to one language: an ISO 639-1 code ("en", "de", ...) as used by the GBIF source, or NA for the untagged NCBI/Open Tree names. NULL (default) searches every language. List the languages present with enrichment_groups("common_names").
output	What to return. "lookup" (default): the lookup table (common name -> scientific name). "result": the matched scientific names run through taxify(), as a taxify_result (with a leading query_common column), so the result pipes straight into the add_*() enrichments.
backbone	Passed to taxify() when output = "result"; NULL (default) uses every installed backbone. Ignored for "lookup".
...	Matching arguments passed to taxify() when output = "result" (e.g. fuzzy, fuzzy_threshold, kingdom, region). Must be named.
verbose	Logical. Default TRUE.

Value

For output = "lookup", a data.frame with one row per (query, scientific match, language):

input_name The common name as supplied.

common_name The vernacular name as stored in the source (its casing, which may differ from input_name).

accepted_name The accepted scientific name.

lang Language tag of the vernacular name (NA for NCBI/Open Tree).

A query with no match contributes no rows. For output = "result", a taxify_result with one row per (query, distinct scientific match), with query_common prepended.

See Also

add_common_names() for the forward direction (scientific -> common), taxify().

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

# The bundled example database maps "example_common_name" to Quercus robur;
# against the full download this is where "pedunculate oak" would resolve.
comm2sci("example_common_name")

# Resolve straight to a taxify_result, ready for add_*() functions
comm2sci("example_common_name", output = "result", backbone = "wfo")

options(old)
```

`downstream` *List all descendants of a taxon down to a target rank*

Description

Returns every accepted taxon at `downto` rank that sits beneath `taxon`, for example every species in an order or family. The parent's rank is detected from the backbone, then descendants are read from the denormalized classification the backbone stores. `children()` is the same query for a genus or family parent.

Usage

```
downstream(
  taxon,
  backbone = NULL,
  downto = "species",
  kingdom = NULL,
  verbose = TRUE
)
```

Arguments

`taxon`	A single higher-taxon name (a genus, family, order, class, phylum, or kingdom).
`backbone`	A single backbone name or a `taxify_backend` object. `NULL` (default) uses the highest-priority installed backbone; name one that stores the higher ranks (e.g. `"col"`) to reach above genus.
`downto`	Target rank of the descendants to return (`"species"` by default), or `"any"` (or `NULL`) for every accepted taxon beneath `taxon` regardless of rank. The parent itself is never returned.
`kingdom`	Optional kingdom (or kingdoms) the parent belongs to, as in `taxify()` (`"plantae"`, `"animals"`, ...). A name can be used in more than one kingdom (*Morus* is both mulberries and gannets); `kingdom` picks one, and also decides the parent's rank when the homonyms sit at different ranks. Without it such a result mixes the kingdoms, with a warning.

verbose Logical. Default TRUE.

Value

A data.frame of accepted descendants, columns: name, authorship, rank, kingdom_group (the coarse kingdom, as in taxify() output), family, genus, taxon_id, parent, parent_rank, backbone. Empty when taxon is not found, its rank is one the backbone does not store as a column (e.g. subfamily, tribe), or it has no descendants at downto.

See Also

children() for a genus or family parent, upstream() for the ancestors, taxify(), synonyms().

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

# Every species the backbone places in the genus
downstream("Quercus", backbone = "col")

options(old)
```

---

enrichment_cols *Browse the columns a published dataset can attach*

Description

Lists the columns available from a published dataset's pre-built .vtr, for choosing which to attach through the functions that accept a cols argument (such as add_gift() and add_floraweb()). Read offline from the local .vtr; the first call may trigger the one-time download.

Usage

```
enrichment_cols(source)
```

Arguments

source Character. An enrichment name (see list_enrichments()).

Value

A data.frame with one row per column: column (the name) and type ("numeric" or "character").

See Also

add_gift(), add_floraweb(), list_enrichments()

Examples

```
old <- options(taxify.data_dir = taxify_example_data())
enrichment_cols("gift")
options(old)
```

`enrichment_groups` *Browse the group values a grouped enrichment can filter on*

Description

Some `add_<source>()` functions attach data per group: GRIIS invasive status by country (`add_griis()`), WCVP native ranges by TDWG region (`add_wcvp()`), vernacular names by language (`add_common_names()`), alien first records by location (`add_alien_first_records()`). This lists the valid group values for such a function, the way `enrichment_cols()` lists a dataset's columns, so a country, region, or language code need not be guessed. Read offline from the local `.vtr` metadata (falling back to the manifest, then a scan of the `.vtr`); the first call may trigger the one-time download.

Usage

```
enrichment_groups(source, verbose = TRUE)
```

Arguments

`source`	Character. A grouped enrichment name (see `list_enrichments()`).
`verbose`	Logical. Print the group column and count. Default `TRUE`.

Value

A character vector of the available group values, sorted. Stops with a pointer to `enrichment_cols()` when `source` is a flat (non-grouped) enrichment, which has no group values.

See Also

`enrichment_cols()`, `list_enrichments()`, `add_griis()`, `add_wcvp()`, `add_common_names()`, `add_alien_first_records()`

Examples

```
old <- options(taxify.data_dir = taxify_example_data())
enrichment_groups("griis")   # ISO country codes GRIIS covers
options(old)
```

euromed_areas *Euro+Med areas and their country codes*

Description

Lists the areas Euro+Med PlantBase states a taxon's status for, with the ISO 3166-1 alpha-2 country code of the areas that are one country. Areas are not countries: some combine several ("Italy, with San Marino and Vatican City"), some are part of one (a region of European Russia, an island group), and the level-2 areas split a combined area into its countries (`Au(A)` Austria and `Au(L)` Liechtenstein under `Au`). Only an area that is exactly one country carries an `iso2`.

Usage

```
euromed_areas()
```

Value

A data.frame with `area_code`, `area_name`, `area_level` and `iso2`.

See Also

`add_euromed_distribution()`

Examples

```
old <- options(taxify.data_dir = taxify_example_data())
euromed_areas()
options(old)
```

export_data *Export a taxify result to file*

Description

Writes a `taxify()` result (with any enrichments) to disk in one of several formats. The default `.vtr` format preserves column types and is fast to re-read with `add_data()`.

Usage

```
export_data(x, path, overwrite = FALSE)
```

Arguments

`x`	A data.frame returned by `taxify()`.
`path`	Character. Output file path. The format is inferred from the extension: `.vtr`, `.csv`, `.tsv`, or `.xlsx`.
`overwrite`	Logical. Overwrite an existing file? Default `FALSE`.

Value

Invisibly returns `path`.

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

result <- taxify(c("Quercus robur", "Pinus sylvestris"))
result |> export_data(tempfile(fileext = ".vtr"))
result |> export_data(tempfile(fileext = ".csv"))
result |> export_data(tempfile(fileext = ".tsv"))

options(old)
```

---

gbif_backmatch *Link GBIF occurrence records back to the names they were requested for*

Description

A request built from `gbif_request()` covers several keys per name, so the records that come back have to be traced to the name that asked for them before they can be counted or grouped.

Usage

```
gbif_backmatch(records, x, verbose = TRUE)
```

Arguments

`records`	A data.frame of occurrence records: what `gbif_request(method = "search")` returns, or a download imported with rgbif's `occ_download_import()`.
`x`	What the keys came from: the object `gbif_request()` returned, a `taxify_ids()` table, or a `taxify()` result.
`verbose`	Logical. Default `TRUE`.

Details

Joining on `taxonKey` alone loses records, silently. GBIF returns the records of a key's descendants as well as its own, and a record identified to a subspecies carries the subspecies key: requesting *Pinus nigra* (5284809) returns records whose `taxonKey` is 5686674, *P. nigra* subsp. *salzmannii*. Those rows match on `speciesKey` and on nothing else. This function therefore tries the record's keys from the most specific outwards and takes the first that is one of the requested keys.

Value

records with three columns added: requested_key (the key that matched), input_name (the name as queried) and accepted_name. Records matching no requested key keep NA in all three. The provenance on x travels with them, so cite() on the result reports the backbone and, for a download, its DOI.

See Also

gbif_request(), taxify_ids().

Examples

```
## Not run:
spp <- c("Pinus nigra", "Quercus robur")
recs <- gbif_request(spp, method = "search", limit = 100)

recs <- gbif_backmatch(recs, recs)
table(recs$input_name, useNA = "ifany")

## End(Not run)
```

---

gbif_fetch *Fetch the records of a submitted GBIF download*

Description

Waits for the download gbif_request() submitted, retrieves it, imports it and returns the records. A request too long for one GBIF query is split across several downloads, and all of them are waited for and stacked, so a sharded request is fetched the same way as a single one.

Usage

```
gbif_fetch(x, backmatch = TRUE, path = tempdir(), verbose = TRUE)
```

Arguments

x	What gbif_request(method = "download") returned.
backmatch	Logical. Attach the queried names with gbif_backmatch(). Default TRUE.
path	Directory to download into. Defaults to a session temp directory, so nothing is written outside it unless path is set.
verbose	Logical. Default TRUE.

Value

A data.frame of occurrence records, carrying the same keys, taxa, taxify_meta and gbif_download attributes as x, so cite() reports the backbone and every download DOI.

See Also

gbif_request(), gbif_backmatch(), cite().

Examples

```
## Not run:
dl <- gbif_request(c("Quercus robur", "Bellis perennis"))
recs <- gbif_fetch(dl)
cite(recs)

## End(Not run)
```

---

gbif_request *Request GBIF occurrence data for a matched name list*

Description

Takes a list of names, matches it against the GBIF backbone and requests occurrence data for every accepted taxon key the matches resolve to. A name GBIF files under more than one key (a homonym published by two authors, or a name held once as accepted and again as a doubtful record) contributes all of its keys, so the request covers the whole matched concept rather than whichever key taxify() reported first.

Usage

```
gbif_request(
  x,
  method = c("download", "search"),
  keys = c("taxon", "pick", "all"),
  limit = 500,
  format = "SIMPLE_CSV",
  dry_run = FALSE,
  ...,
  verbose = TRUE
)
```

Arguments

x	A character vector of names, or a taxify() result. A character vector is matched against the GBIF backbone first.
method	"download" (authenticated, no cap, citable) or "search" (unauthenticated, capped). See Choosing a method.
keys	Which of a name's GBIF keys to request, by their role in taxify_ids(). "taxon" (default) requests the key taxify() picked and every other record of the same taxon (a doubtful or duplicate record of the name), and reports the keys of other taxa a homonym's synonym records point to. "pick" requests the

	picked key alone. "all" requests every key, other taxa included. The keys left out are named in the verbose line.
limit	Records per key for method = "search". Ignored by "download".
format	Download format for method = "download", passed to rgbif. Ignored by "search".
dry_run	Logical. If TRUE, return the keys without contacting GBIF.
...	Matching arguments passed to taxify() (fuzzy, kingdom, region, ...); only when x is a character vector, and must be named.
verbose	Logical. Default TRUE.

Details

The keys themselves are read off taxify_ids(), which resolves each one against the backbone and carries its occurrence count; call it directly to see what a request would cover before sending it, or pass dry_run = TRUE here for the keys alone.

Value

With dry_run = TRUE, an integer vector of GBIF taxon keys, which prints as the keys alone. With method = "download", the object rgbif's occ_download() returns (the download key, to be passed to occ_download_wait()). With method = "search", a data.frame of occurrence records, empty if none matched. In every case the keys and the taxify_ids() table behind them are attached as the keys and taxa attributes, along with the match's taxify_meta; a download also carries its key as gbif_download, which is what lets cite() report the download's DOI beside the backbone.

Choosing a method

method = "download" submits an asynchronous GBIF download. It has no record cap and yields a citable DOI, which is what GBIF asks for in published work, and it needs a GBIF account: set GBIF_USER, GBIF_PWD and GBIF_EMAIL in ~/.Renviron. The call returns as soon as the request is queued; wait for it and fetch it with rgbif's occ_download_wait() and occ_download_get().

method = "search" sends unauthenticated searches and returns the records directly. It needs no account, and the GBIF search API caps what it will page through, so it suits a look at a handful of taxa rather than a checklist-wide pull.

See Also

taxify_ids() for the keys and their occurrence counts, taxify() for the matching itself.

Examples

```
## Not run:
# A checklist, matched and requested in one call. Needs the full GBIF
# backbone, so it cannot run on a check machine.
spp <- c("Quercus robur", "Bellis perennis", "Morus alba")

# What would be requested, without contacting GBIF:
gbif_request(spp, dry_run = TRUE)
```

```
# An unauthenticated search:
recs <- gbif_request(spp, method = "search", limit = 50)

# A citable download (needs GBIF_USER / GBIF_PWD / GBIF_EMAIL):
dl <- gbif_request(spp, method = "download")
rgbif::occ_download_wait(dl)
rgbif::occ_download_get(dl)

## End(Not run)
```

`gift_traits` *Browse the bundled GIFT trait columns*

Description

Returns the species-level trait columns available from the bundled GIFT dataset, for picking which to attach in `add_gift()`. Read offline from the local `.vtr` (downloaded or built once); the first call may trigger that one-time download.

Usage

```
gift_traits()
```

Value

A data.frame with one row per trait column:

column The `gift_<trait>` column name.

type `"numeric"` or `"character"`.

See Also

`add_gift()`, `enrichment_cols()` for the same listing on any enrichment.

Examples

```
old <- options(taxify.data_dir = taxify_example_data())
gift_traits()
options(old)
```

id2name *Resolve backbone IDs to names*

Description

Looks up one or more backbone taxon IDs in a backbone and returns the name, rank, classification, and accepted-name resolution for each. The inverse of the `taxon_id` / `accepted_id` columns `taxify()` emits.

Usage

```
id2name(id, backbone = NULL, verbose = TRUE)
```

Arguments

`id`	A vector of backbone IDs (e.g. GBIF keys, ITIS TSNs, WoRMS AphiaIDs), matched as text against the backbone's `taxon_id`. Numeric IDs (as read from a CSV) are written out in full, never in scientific notation, and must be whole numbers. IDs are backbone-specific, so name the `backbone` they came from.
`backbone`	A single backbone name (e.g. `"col"`, `"gbif"`) or a `taxify_backend` object. `NULL` (default) uses the highest-priority installed backbone.
`verbose`	Logical. Default `TRUE`.

Value

A data.frame with one row per input ID (in input order), columns:

id The queried ID.

name Canonical name for that ID (`NA` if the ID is not in the backbone).

authorship Authorship of the name.

rank Taxonomic rank.

is_synonym Logical. Is this ID a synonym?

accepted_name The accepted name the ID resolves to (equals `name` when the ID is itself accepted).

family Family.

genus Genus.

backbone Backend used.

IDs not found in the backbone yield a row with `NA` name columns, so the output stays aligned one-to-one with the input.

See Also

`taxify()` for name -> ID, `synonyms()`, `add_classification()`.

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

# Round-trip: resolve a name, then look its ID back up
r <- taxify("Quercus robur", backbone = "col")
id2name(r$taxon_id, backbone = "col")

options(old)
```

`inspect` *Inspect a name list for probable typos and other anomalies*

Description

A quality-control pass over a name list. By default `inspect()` does not match names against backbones: on a plain character vector it runs the checks that need no matching (the genus register and the rest of the batch) and is fast and offline. To also surface the match-based anomalies (`typo`, `synonym`, `ambiguous`, `geographic`), either set `backbones = TRUE` (matches against every installed backbone, listed in the report) or match yourself first and inspect the result (`taxify(x) |> inspect()`). Either way it returns only the rows that look anomalous, each labelled with what stands out and, where known, the name to use instead.

Usage

```
inspect(
  x,
  backbones = FALSE,
  region = NULL,
  coords = NULL,
  range = c("present", "native", "introduced"),
  min_tier = c("note", "review", "unresolved"),
  verbose = TRUE
)
```

Arguments

`x`	A character vector of names, or a `taxify_result` from `taxify()`.
`backbones`	Logical. When `x` is a character vector, `TRUE` matches it against every installed backbone, in the order of the default fallback chain of `taxify()`, so the match-based labels are available; `FALSE` (default) runs the register and list checks only, with no matching. The backbones used are printed in the report header. Ignored when `x` is already a `taxify_result` (it was matched already).
`region, coords, range`	Geographic constraint for the `geographic` / `out_of_range` checks, as in `taxify()`. These act on matched names: a `taxify_result`, or a character vector with

backbones = TRUE. On a character vector inspected without matching there is nothing matched to place, so they have no effect and are not resolved. A region / coords that resolves to no known region is named under not checked rather than treated as a region no species occurs in.

min_tier — Lowest tier to report: "note" (default, everything), "review", or "unresolved".

verbose — Logical. Print progress messages. Default TRUE.

Details

Checks that need no matching (run on a character vector or a result):

unknown The genus is not in the genus register, the union of the genera of every backbone, so no backbone recognises it. The strong "probably not a real name" signal.

near_duplicate A near-twin of a more frequent name in the same list (small edit distance), so probably a misspelling of it. Computed from the list alone, so it catches typos in names no backbone contains.

outlier_group The name's kingdom group (from the register) is a tiny minority of an otherwise group-coherent list: the lone animal or fungus among plants, typically a cross-kingdom homonym typo.

Checks read from a taxify() result (only present when the input is one):

typo Resolved only after fuzzy correction (match_type = "fuzzy"): the input most likely contains a spelling error; suggestion is the name.

rank_fallback An infraspecific name the backbone does not carry, resolved to its species (match_type = "rank_fallback").

ambiguous A name the backbone files under more than one accepted taxon (n_ids > 1) that is not itself the accepted name; the reason lists the accepted IDs.

geographic The matched species is real but has no WCVP record in the declared region / coords (vascular plants only).

out_of_range No region declared, yet the matched species' range falls outside the list's main TDWG continents (skipped for globally spread lists).

case Resolved only after ignoring case (match_type = "exact_ci").

synonym The input is an outdated synonym, or a name the backbone keeps unplaced whose basionym it places (match_type = "basionym"); suggestion is the current accepted name.

Rows with no anomaly are dropped.

Each row gets a tier describing what it needs, not how bad it is: unresolved (no usable name: act on it), review (a name is there but its identity is uncertain: verify it), or note (correct, optional cleanup). unknown is unresolved; the identity-uncertain labels are review; case and synonym are note. An anomaly may be intended, so the tier is a triage hint, not a verdict.

The list-context labels (near_duplicate, out_of_range, outlier_group) judge a name against the rest of the batch, so they cannot apply to a single name: inspect() on one name warns and reports only the per-name labels. The register checks (unknown, and the register-derived outlier_group) need the genus register installed, and the range checks need the WCVP range data; a check that could not run is named in the report header under not checked, so an empty report is never mistaken for a clean one.

Value

A `taxify_inspection` data.frame (one row per anomalous name, ordered most-notable first) with columns `input_name`, `suggestion` (the name to use instead, or `NA`), `anomalies` (`|`-joined labels), `tier` (ordered factor `note` < `review` < `unresolved`), `reason`, `fuzzy_dist`, and `backbone`. Zero rows means nothing stood out among the checks that ran; the header's `not checked` line names any that could not.

See Also

`taxify()`, `taxify_regions()`

Examples

```
old <- options(taxify.data_dir = taxify_example_data())

# On raw names: register + list checks (no matching)
inspect(c("Quercus robur", "Bogusus fakus", "Carexus mysteriosa",
          "Carexus mysteriosa", "Carexus mysteryosa"))

# Opt in to matching to also get typos, synonyms, ambiguity
inspect(c("Quercus robur", "Quercus robus"), backbones = TRUE)

# Or match yourself and inspect the result
taxify(c("Quercus robur", "Quercus robus")) |> inspect()

options(old)
```

---

install_backbones *Install taxonomic backbones for offline matching*

Description

Downloads the pre-built `.vtr` for each named backbone into the taxify data directory (`taxify_data_dir()`), so subsequent `taxify()` calls match against them offline. Called without arguments it installs the default set (COL, GBIF, ITIS, about 1.1 GB); name backbones to install a different set, add a backbone, or refresh to the latest release. Calling it is the consent to download: other functions that reach a backbone not on disk ask first in an interactive session and stop with an error otherwise, unless `options(taxify.auto_download = TRUE)` (or the `TAXIFY_AUTO_DOWNLOAD` environment variable) allows the download. `taxify_clean()` removes backbones again.

Usage

```
install_backbones(backbones = NULL, verbose = TRUE)
```

Arguments

backbones	Character vector of backbone names (see list_backbones()). NULL (default) installs taxify's first-run set: COL, GBIF, and ITIS.
verbose	Logical. Default TRUE.

Details

An installed backbone is compared against the manifest by the same check taxify() runs once per session: by content id where both sides record one, else by version. A backbone whose build differs from the one the manifest serves is replaced with the current release; one that is already current is left as it is. A build pinned by taxify_pin(), taxify_restore() or taxify_download(content_id =) is never refreshed, and a message says so; taxify_pin(backbone, pin = FALSE) releases it. With options(taxify.offline = TRUE) nothing is compared and installed backbones are kept.

Value

Invisibly, the backbones now installed (those that downloaded successfully, or were already on disk), in priority order. A backbone whose refresh failed stays in the result with a warning, since its previous build is still usable.

See Also

list_backbones() for the full set with sizes, taxify().

Examples

```
## Not run:
# Pre-install a marine-focused set before matching:
install_backbones(c("col", "worms"))

## End(Not run)
```

---

list_backbones *List supported taxonomic backbones*

Description

Returns every taxonomic backbone taxify() can match against, its taxonomic scope, and (once the manifest is reachable) its current version, name count, download size, and whether it is already installed locally. The counterpart to list_enrichments() for the backbone side.

Usage

```
list_backbones(verbose = TRUE)
```

Arguments

verbose	Logical. Default TRUE.

Value

A data.frame with columns: `name`, `title` (the source's full name), `scope`, `n_names`, `size_mb`, `version`, `source_date`, `installed`, `code`, `source`. `code` is the nomenclature the backbone's names follow: `"taxon"` for the taxonomic backbones, `"syntaxon"` for a backbone of vegetation units (`"evc"`, the EuroVegChecklist), which is matched only when named in `taxify(backbone = )`. `n_names`, `size_mb`, and `version` are `NA` for any backbone the manifest does not yet describe or when the manifest cannot be fetched offline.

`version` is the release that packaged the backbone; `source_date` is the date of the upstream data, which can be much earlier. The GBIF backbone is the case that matters: GBIF froze it at 2023-08-28 and has said it will not be updated again, so a current release tag there carries a treatment three years older than the tag suggests. `source_date` is an ISO date, or `YYYY-MM` / `YYYY` where the source dates its release no finer, and `NA` for a backbone whose upstream date has not been recorded.

See Also

`list_enrichments()`, `list_traits()`, `taxify_databases()`.

Examples

```
## Not run:
list_backbones()

## End(Not run)
```

---

list_enrichments *List available enrichments*

Description

Returns a summary of all published datasets available in the taxify manifest, including version, row count, whether the dataset is static, and which trait columns are provided.

Usage

```
list_enrichments(verbose = TRUE)
```

Arguments

`verbose`	Logical. Default `TRUE`.

Value

A data.frame with columns: `name`, `version`, `nrow`, `static`, `trait_cols` (comma-separated), and `source_url`.

Examples

```
## Not run:
list_enrichments()

## End(Not run)
```

`list_traits` *List the traits available to add_trait()*

Description

Returns the traits that `add_trait()` can attach across sources, with their kind, canonical unit, and the number and names of contributing sources.

Usage

```
list_traits()
```

Value

A data.frame with one row per trait:

trait The trait name to pass to `add_trait()`.

label Human-readable label.

kind `"numeric"` or `"categorical"`.

unit Canonical unit for numeric traits, `NA` for categorical.

n_sources Number of sources providing the trait.

sources Comma-separated source names.

See Also

`add_trait()`, `trait_info()`

Examples

```
list_traits()
```

lookup_genus *Look up a genus in the register*

Description

Returns the register row for the given genus, or NULL if not found. Auto-loads the register on first call.

Usage

```
lookup_genus(genus)
```

Arguments

genus	Character scalar. The genus name to look up.

Value

A one-row data.frame, or NULL if the genus is not in the register (including NA).

lowest_common *Lowest common taxon of a set of names*

Description

Resolves the names and reports the deepest Linnaean rank at which they all share one classification value: their most recent common ancestor in the backbone's hierarchy. Two congeners return their shared genus; two plants in different families return a shared order or class; unrelated taxa share only a kingdom (or nothing).

Usage

```
lowest_common(x, backbone = NULL, ..., verbose = TRUE)
```

Arguments

x	Character vector of names (two or more), or a taxify() result.
backbone	Backend passed to taxify() when x is raw names. NULL (default) uses every installed backbone. Ignored when x is a result.
...	Matching arguments passed to taxify() when x is raw names (e.g. fuzzy, fuzzy_threshold, kingdom, region). Must be named.
verbose	Logical. Default TRUE.

Value

A one-row data.frame with columns `rank`, `name` (the shared taxon), and `n_taxa` (how many resolved names went into the comparison). `rank` and `name` are `NA` when the taxa share nothing (e.g. different kingdoms, or only one resolved).

See Also

`class2tree()`, `add_classification()`, `taxify()`.

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

lowest_common(c("Quercus robur", "Quercus petraea"))

options(old)
```

---

parse_name *Parse taxonomic names into their structural parts*

Description

Decomposes each name into genus, specific epithet, infraspecific rank and epithet, authorship, hybrid status, and any open-nomenclature qualifier, without matching against a backbone. Where `taxify()` cleans a name and then resolves it, `parse_name()` returns the cleaning step itself (the parts of the name) for callers who need the decomposition rather than a match (the role `rgbif::name_parse()` or gnparser fill). The same cleaning pipeline the matcher uses drives the parse, so a name breaks apart the way it matches.

Usage

```
parse_name(x, code = c("taxon", "syntaxon"))
```

Arguments

`x`	Character vector of taxonomic names.
`code`	Character. The nomenclature the names follow: `"taxon"` (default; botanical and zoological names) or `"syntaxon"` (vegetation units under the International Code of Phytosociological Nomenclature, the names a syntaxon backbone such as `"evc"` matches; see `taxify()`).

Value

For code = "taxon", a data.frame with one row per input name and columns:

input_name The original name as supplied.

genus The genus (first token), or a single-letter initial for an abbreviated genus ("Q. robur"). NA for an unresolvable hybrid formula.

specific_epithet The specific epithet, or NA for a bare genus.

infrasp_rank The infraspecific rank marker ("subsp.", "var.", "f.", ...) when the name has one, else NA.

infrasp_epithet The infraspecific epithet, else NA.

authorship The authorship the name carries (trailing form, e.g. "(L.) H.Karst."), or NA.

qualifier Any open-nomenclature / uncertainty qualifier ("cf.", "aff.", "sp.", "agg.", "s.l.", ...), canonicalized, or NA. Infraspecific rank markers are reported in infrasp_rank, not here.

is_hybrid Logical. Was a hybrid marker detected?

hybrid_type "nothogenus", "nothospecies", "formula", or NA.

rank "genus", "species", "infraspecies", "hybrid_formula", or NA for an empty input.

canonical The cleaned name used for matching (genus plus epithets and any infraspecific rank marker in its canonical spelling, qualifiers and authorship removed, hybrid sign dropped). NA for a hybrid formula, which is not a single taxon.

For code = "syntaxon", a data.frame with one row per input name and columns:

input_name The original name as supplied.

code A national or database code written in front of the name ("TDA - Arrhenatherion elatioris", "41BB03 - Tilio-Carpinetum"), or NA.

name The syntaxon name without author or annotations, the form the matcher looks up. Where a record names several units, the most specific rank-labelled one ("all. X, ord. Y"), else the first.

authorship The author citation ("(Oberd. 1957) Passarge 1964"), or NA.

rank The rank the name's suffix denotes: "class" (-etea), "order" (-etalia), "alliance" (-ion), "suballiance" (-enion), "association" (-etum), "subassociation" (a trailing -etosum), "subclass", "suborder", or NA.

qualifier "?" for a doubtful assignment, "cf.", "s.l.", "s.str.", or NA.

name_status Nomenclatural annotations after the citation ("nom. inval. (5a)", "(syntax.syn.)", "(Regionalverband)"), or NA.

See Also

taxify() to match a name; parse_name() exposes the same internal cleaning pipeline the matcher runs.

Examples

```
parse_name(c("Quercus robur L.",
             "Poa annua var. annua",
             "Q. robur",
             "Pinus cf. sylvestris",
             "Salix alba x Salix fragilis"))

parse_name(c("Caricenion gracilis (Neuhausl 1959) Oberd. et al. 1967",
             "TDA - Arrhenatherion elatioris",
             "Conioselinion tatarici Golub et al. 2003 nom. inval. (5a)"),
           code = "syntaxon")
```

`reconcile` *Reconcile a checklist against a backbone's current treatment*

Description

Classifies each name in a list against what a backbone currently accepts, so a checklist assembled at one time can be checked against the taxonomy as it stands now. Where `taxify()` returns the resolved match per name, `reconcile()` adds the editorial verdict (unchanged, now a synonym, a misspelling, ambiguous, or unresolved) and flags names that merge onto a single accepted taxon.

Usage

```
reconcile(x, backbone = NULL, ..., verbose = TRUE)
```

Arguments

`x`	Character vector of names (a checklist).
`backbone`	Character vector of backbone names or a `taxify_backend` object, passed to `taxify()`. `NULL` (default) uses every installed backbone.
`...`	Matching arguments passed to `taxify()` (e.g. `fuzzy`, `fuzzy_threshold`, `kingdom`, `region`). Must be named.
`verbose`	Logical. Default `TRUE`.

Value

A data.frame with one row per input name, columns:

input_name The name as supplied.

accepted_name The name it resolves to now (`NA` if unresolved).

status One of: `"unchanged"` (resolves to itself, still accepted), `"synonym"` (now a synonym of a different accepted name, including a name the backbone keeps unplaced but resolves through its basionym), `"misspelling"` (resolved by fuzzy/abbrev match to a corrected spelling), `"rank_fallback"` (an infraspecific name the backbone does not carry; `accepted_name` is

the accepted name of its species), "ambiguous" (a name the backbone files under several accepted taxa, n_ids > 1, that does not resolve to itself as accepted; accepted_name is the one it picked, and taxify_ids() lists the others), "unresolved" (no match).

is_synonym Logical, from taxify().

match_type The taxify() match type.

n_ids Distinct accepted taxa the backbone files the name under, from taxify(). Above 1 on an "unchanged" row, the name is accepted under its own spelling and a homonym record also exists.

merged Logical. TRUE when two or more input names resolve to this same accepted name (a many-to-one collapse).

merged_with |-joined other input names sharing this accepted name, or NA.

backbone Backend that matched (NA if unresolved).

See Also

taxify(), taxify_ids() to expand the ambiguous rows, synonyms().

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

# Quercus pedunculata is now a synonym of Quercus robur; "Quercus robus" is a
# misspelling of it; a bad name is unresolved.
reconcile(c("Quercus robur", "Quercus pedunculata", "Quercus robus",
            "Notagenus imaginus"),
          backbone = "wfo", verbose = FALSE)

options(old)
```

---

sci2comm *Resolve scientific names to common (vernacular) names*

Description

The forward direction of comm2sci(): given a scientific name, return the common name(s) it is known by. Reads the bundled common_names enrichment (GBIF, NCBI, and Open Tree vernaculars) offline; the first call may trigger the one-time download. Where add_common_names() attaches vernaculars to a taxify() result as columns, sci2comm() takes bare names and returns the long lookup table, one row per (name, vernacular). A name maps to many common names (across languages and synonyms), so a query can carry several rows.

Usage

```
sci2comm(x, lang = NULL, resolve = TRUE, backbone = NULL, ..., verbose = TRUE)
```

Arguments

x	Character vector of scientific names.
lang	Character. Restrict to one language: an ISO 639-1 code ("en", "de", ...) as used by the GBIF source, or NA for the untagged NCBI/Open Tree names. NULL (default) returns every language. List the languages present with enrichment_groups("common_names").
resolve	Logical. When TRUE (default), each input is run through taxify() first so a synonym or misspelling reports its accepted taxon's vernaculars. When FALSE, the input name is looked up verbatim (faster, offline; use when the names are already accepted).
backbone	Passed to taxify() when resolve = TRUE; NULL (default) uses every installed backbone. Ignored when resolve = FALSE.
...	Matching arguments passed to taxify() when resolve = TRUE (e.g. fuzzy, fuzzy_threshold, kingdom, region). Must be named.
verbose	Logical. Default TRUE.

Value

A data.frame with one row per (query, vernacular):

input_name The scientific name as supplied.

accepted_name The accepted name looked up (equals input_name when resolve = FALSE or the input_name was already accepted).

common_name A vernacular name.

lang Language tag (NA for NCBI/Open Tree).

name_rank Rank of the name among the taxon's names in that language, 1 being the one most source records give (the name add_common_names() reports).

A query with no vernacular contributes no rows.

See Also

comm2sci() for the reverse direction (common -> scientific), add_common_names() to attach vernaculars to a taxify() result.

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

# The bundled example maps Quercus robur -> "example_common_name" (en + de)
sci2comm("Quercus robur", resolve = FALSE)

options(old)
```

`synonyms` *List the synonyms of a name*

Description

The reverse of the forward resolution `taxify()` does: each input name is resolved to its accepted taxon, then every synonym that points to that accepted taxon in the backbone is returned. Useful for auditing which historical names collapse onto a current name.

Usage

```
synonyms(x, backbone = NULL, ..., verbose = TRUE)
```

Arguments

`x`	Character vector of names (accepted names or synonyms; each is resolved to its accepted taxon first).
`backbone`	A single backbone name (e.g. `"wfo"`) or a `taxify_backend` object. `NULL` (default) uses the highest-priority installed backbone.
`...`	Matching arguments passed to `taxify()` when resolving `x` (e.g. `fuzzy`, `fuzzy_threshold`, `kingdom`, `region`). Must be named.
`verbose`	Logical. Default `TRUE`.

Value

A data.frame with one row per (distinct input name, synonym); a name supplied more than once is reported once. Columns:

input_name The queried name.

accepted_name The accepted name the query resolved to.

synonym A synonym of that accepted taxon.

authorship Authorship of the synonym.

rank Rank of the synonym.

taxon_id Backend ID of the synonym.

backbone Backend used.

Names that resolve to an accepted taxon with no synonyms contribute no rows.

See Also

`taxify()` for the forward direction, `children()` to list the accepted taxa within a genus or family.

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

# Amphibolurus vitticeps is a synonym of Pogona vitticeps
synonyms("Pogona vitticeps", backbone = "reptiledb")

options(old)
```

`taxify` *Match taxonomic names against local backbone databases*

Description

Matches a vector of taxonomic names against locally stored Darwin Core backbone databases. Returns a data.frame with one row per input name containing the matched name, accepted name, taxonomic hierarchy, and match quality information.

Usage

```
taxify(
  x,
  backbone = NULL,
  fuzzy = TRUE,
  fuzzy_threshold = 0.2,
  fuzzy_method = c("dl", "levenshtein", "jw"),
  aggregates = c("preserve", "collapse"),
  region = NULL,
  coords = NULL,
  range = c("present", "native", "introduced"),
  kingdom = NULL,
  mode = c("fallback", "wide", "agreement"),
  verbose = TRUE
)
```

Arguments

`x`	Character vector of taxonomic names.
`backbone`	Character vector of backbone names (e.g., `"wfo"`, `"col"`, `"gbif"`) or a single `taxify_backend` object. Several are tried in order as a fallback chain. `NULL` (default) uses every installed backbone in priority order, offering the default set (COL, GBIF, ITIS) when none is installed; override the priority with `options(taxify.backbone_priority = ...)` and the first-run set with `options(taxify.default_bac = ...)`.
`fuzzy`	Logical. Enable fuzzy matching for names that fail exact match. Default `TRUE`.

`fuzzy_threshold`

Numeric. Maximum allowed distance for fuzzy matches. Two modes depending on the value:

- **Fractional** (`0 < fuzzy_threshold < 1`): normalized distance (edits / max name length). Default `0.2` is about 1 edit per 5 characters.
- **Integer** (`fuzzy_threshold >= 1`): maximum raw edit count, e.g. `fuzzy_threshold = 2L` allows at most 2 insertions/deletions/substitutions regardless of name length. Not supported for `fuzzy_method = "jw"`.

`fuzzy_method` Character. One of `"dl"` (Damerau-Levenshtein, default), `"levenshtein"`, or `"jw"` (Jaro-Winkler).

`aggregates` Character. How to treat species aggregates (names with an `agg.` / `s.l.` qualifier). `"preserve"` (default) keeps the aggregate as its own concept: it matches the backbone's aggregate taxon (`"<binomial> aggr."`) where one exists, otherwise falls back to the binomial. The output's `aggregate_fallback` column records which happened for each aggregate: `TRUE` where it fell back to the binomial, `FALSE` where it resolved to the dedicated aggregate taxon. Only Euro+Med and WoRMS carry aggregate taxa, so preserve falls back for the other backbones. `"collapse"` strips the marker and matches the binomial species, the way any non-aggregate name is matched. Either way the qualifier is recorded in the `qualifier` column.

`region` Region(s) to constrain fuzzy matching to, or `NULL` (default) for no geographic constraint. Botanical (WCVP, vascular plants): TDWG Level 3 codes (`"BGM"`, `c("BGM", "GER")`) or region names at any level, matched case- and accent-insensitively against the bundled WGSRPD crosswalk: a Level 3 name (`"Belgium"`), a Level 2 region (`"Middle Europe"`), or a Level 1 continent (`"Europe"`, which expands to all its codes). Marine (only when the `marine_distribution` asset is installed): a MEOW ecoregion `ECO_CODE`, or an ecoregion / province / realm name (a province or realm expands to its member ecoregions). See `taxify_regions()` for both lists. When set, **fuzzy** candidates are restricted to species with range records in the region(s); exact matches are always kept. The filter only narrows genuinely ambiguous fuzzy candidates: a candidate is dropped only when the same input name has another candidate that is in-region or has no range data, so a match in a group with no range coverage is never affected and a name whose only candidate is out-of-region is still returned.

`coords` Coordinates to constrain fuzzy matching to, mapped to region codes by point-in-polygon and unioned with `region`. Points are tested against the WGSRPD botanical boundaries (yielding TDWG codes) and, when the marine asset is installed, the MEOW ecoregion boundaries (yielding `ECO_CODE`s), so a coastal point can resolve to both. A single `c(lon, lat)` pair, a matrix/data.frame of longitude/latitude columns (named `lon`/`lat` or `x`/`y`, else the first two columns as lon, lat), or a point-geometry spatial object (an **sf**/`sfc` object or a **terra** `SpatVector`, reprojected to longitude/latitude automatically). `NULL` (default) for none. Each boundary file is downloaded once and cached; coordinate lookup needs that download (or a prior cache). The point-in- polygon test uses **terra** or **sf** when installed, otherwise a native fallback; force the engine with `options(taxify.pip_engine = "terra" | "sf" | "native")`.

`range` Character. Which range statuses count as in-region when `region` or `coords`

is set. "present" (default) accepts any record (native, introduced, extinct, or unknown status), the right choice for name disambiguation. "native" accepts only native records, "introduced" only introduced (alien) records; both fold an ecological filter into matching and are for callers who want that. Ignored when no region is set.

kingdom — Character. Restrict matches to one or more kingdoms, to disambiguate a name shared across kingdoms (a *Prunella* that is both a bird and a plant, an *Oenanthe* that is both). NULL (default) applies no constraint. Accepts a kingdom name or a common alias, case-insensitively: "animals"/"Animalia"/"Metazoa", "plants"/"Plantae", "fungi", "bacteria", "archaea", "chromista", "protozoa", "viruses". A matched taxon is kept only when its kingdom is the requested one (or is unknown, which is never rejected); with the default multi-backbone fallback, a name a backbone resolves into the wrong kingdom is passed on to the next backbone, so the in-kingdom treatment wins. The kingdom is read from the backbone where it stores one (COL, ITIS, NCBI, OTT, WoRMS); for a backbone that does not (WFO, GBIF), it falls back to the genus register's kingdom, which cannot split a genus that is itself homonymous across kingdoms; name a kingdom-appropriate backbone for those.

mode — Character. How to combine results when backbone names more than one backbone. "fallback" (default) is the fallback chain described above: one answer per name, from the first backbone that matched. "wide" and "agreement" instead consult **every** backbone for every name and report how they compare, so a backbone disagreement (see the *Backbone-specific accepted names* section) is visible in one call rather than by querying each backbone by hand. Both return a strict superset of the "fallback" result (the same standard columns, with accepted_name still the fallback pick, so the frame still pipes into the add_*() enrichments) plus:

- "wide": one accepted_<backbone> column per backbone and a logical all_agree.
- "agreement": n_backbones_matched, n_distinct_accepted, and all_agree.

all_agree is TRUE/FALSE when at least two backbones matched the name and NA when fewer than two did (nothing to compare). Ignored (with a message) when only one backbone is given, since there is nothing to compare.

verbose — Logical. Print progress messages. Default TRUE.

Details

By default taxify() matches against **every installed backbone**, tried in priority order as a fallback chain (the COL syntheses, then the domain authorities, then the broad aggregators). The chain is staged by match quality: every backbone is asked for an exact match first, and only the names still unresolved go round again for a fuzzy one. A name is therefore resolved by the highest-priority backbone that matches it *at the best quality any backbone reaches*, so a near neighbour in an early backbone does not settle a name a later backbone holds exactly. Names matched earlier are not re-matched later. On a fresh setup with nothing installed yet, install_backbones() installs the default set (COL, GBIF, ITIS); a first call without it asks before downloading in an interactive session and stops otherwise. Name a backbone (or several) explicitly to match only against that one, or those in that order.

Value

A data.frame with one row per input name and the following columns:

input_name The original name as provided.

matched_name Full name in the backbone that matched. For an unresolved hybrid formula (`match_type = "hybrid_formula"`) it holds the input-parent cross (e.g. `"Salix alba x Salix fragilis"`) when both parents resolve, else `NA`.

accepted_name Resolved accepted name (equals `matched_name` if not a synonym). For a hybrid formula it holds the accepted-parent cross (both parents resolved), else `NA`.

taxon_id Backend-specific ID of the matched name.

accepted_id ID of the accepted name.

rank Taxonomic rank (species, subspecies, genus, etc.).

family Family name.

genus Genus name.

epithet Specific epithet.

authorship Authorship of the matched name.

accepted_authorship Authorship of the accepted name. For a synonym this is the author of the resolved accepted name, not the synonym's own author, so `accepted_name` and `accepted_authorship` together form the accepted name's full citation.

is_synonym Logical. Does the matched name resolve to a different accepted taxon? `TRUE` for a synonym record, and for an unplaced record resolved through its basionym (`match_type = "basionym"`).

taxonomic_status The matched record's own status as the backbone writes it (`"ACCEPTED"`, `"SYNONYM"`, WFO's `"UNCHECKED"`, COL's `"PROVISIONALLY ACCEPTED"`, ...), so a name the backbone holds without having placed it can be told from an accepted one. `NA` when nothing matched.

is_hybrid Logical. Was a hybrid marker detected in the input?

hybrid_type `"nothogenus"` (`"x Cupressocyparis leylandii"`), `"nothospecies"` (`"Quercus x hispanica"`), `"formula"` (`"Salix alba x Salix fragilis"`), or `NA` for a non-hybrid. Nothogenus and nothospecies resolve to a single backbone taxon in the usual columns. A formula resolves that way only where the backbone stores the cross; otherwise `match_type` is `"hybrid_formula"`, the ID, rank and classification columns are `NA`, and `matched_name` / `accepted_name` name the cross by its parents when both parents resolve. The parent binomials, and their accepted names, are added on demand by `add_hybrid_info()`.

qualifier Canonical taxonomic qualifier found in the input name (`"cf."`, `"aff."`, `"agg."`, `"s.l."`, `"s.str."`, `"sp."`, ...), or `NA`. Spelling variants are folded to one token (`"aggr."`, `"agg"` and `"sensu lato"` all map to `"agg."`/`"s.l."`).

qualifier_position `"genus"` when the qualifier leads the name and qualifies the whole name (e.g. `"Cf. Pinus sylvestris"`), `"species"` when it qualifies the species (inline `cf.` or trailing `agg.`), `NA` when there is no qualifier.

aggregate_fallback Logical. For an aggregate query under `aggregates = "preserve"`: `FALSE` when it resolved to the backbone's dedicated aggregate taxon, `TRUE` when no such taxon existed and it fell back to the nominal binomial. `NA` for non-aggregate queries and under `aggregates = "collapse"`, where the collapse is explicit.

match_type One of "exact", "exact_ci", "fuzzy", "abbrev" (an abbreviated genus such as "Q. robur" resolved via genus initial plus epithet), "hybrid_formula" (a two-parent cross the backbone does not store; the ID, rank and classification columns are NA, matched_name / accepted_name name the cross by its parents when both resolve, and add_hybrid_info() materializes the parents into the hybrid_parent_* columns), "rank_fallback" (an infraspecific name no backbone carries, resolved to its species: the ID, rank and name columns describe the species), "basionym" (a record the backbone keeps unplaced, resolved to the taxon under which it places the record's basionym: matched_name, taxon_id and taxonomic_status describe the unplaced record, the accepted_* columns that taxon; needs a backbone built with its basionym links), or "none".

fuzzy_dist Normalized string distance (0–1), NA if exact.

n_ids Integer. How many distinct accepted taxa the backbone files the matched name under, across every record of it (accepted, doubtful, unplaced, synonym). 1 for a name with one accepted ID; more for a homonym or a name the backbone holds twice, e.g. GBIF's *Karwinskia mollis*, kept both as the accepted Schltdl. name and as a doubtful Standl. one. NA when nothing matched.

accepted_ids Character. Those accepted IDs, |-joined, the one in accepted_id first (accepted_id itself for a name with one). Both columns are in every result; printing leaves them out when no row has more than one ID. List them with their names, status and GBIF occurrence counts through taxify_ids().

backbone Which backbone was used (e.g., "wfo", "col", "gbif").

backbone_version Backend name, version, and download date (e.g., "wfo:2024-12 (2026-04-01)"). Useful for reproducibility.

kingdom_group Coarse kingdom-level group of the matched genus, from the bundled genus register (used for cross-kingdom disambiguation); NA when the genus is not in the register.

taxon_group Broad taxonomic group of the matched genus, from the genus register; NA when unavailable.

life_form Life-form classification of the matched genus, from the genus register's family-based lookup; NA when unavailable.

Names with several accepted IDs

A backbone can file one name under several accepted taxa: a homonym (the same binomial published by two authors), or a name held twice, once accepted and once as a doubtful or duplicate record. accepted_id holds one of them; accepted_ids and n_ids hold all of them, and taxify_ids() lists them one row per ID with name, authorship, status and, for GBIF, occurrence count.

Which one accepted_id holds is decided by fuzzy distance first, then, on a backbone that carries occurrence counts (GBIF), by whether any GBIF occurrence records are filed under a key that keeps the name as its own concept (accepted, doubtful or unplaced), then by taxonomic status (accepted before doubtful or unplaced before synonym), rank and epithet, and only then by the number of records, before the remaining tiebreaks. An empty accepted key thus loses to a doubtful key of the name that has the data, while a synonym record, whose accepted ID is another taxon, never displaces the name's own key; it stays listed in accepted_ids.

When any input resolves to more than one accepted ID, taxify() issues one warning per call, of class taxify_multiple_ids, naming the count and a few examples. Silence it with options(taxify.warn_multiple_ids = FALSE) or catch it by class.

Backbone-specific accepted names

Each backbone is an independent taxonomy, and they can legitimately disagree on which name is accepted and which is a synonym. `taxify()` returns the matched backbone's own current treatment; it does not reconcile backbones against each other by voting (a consensus would regress toward the most conservative treatment across backbones that copy one another). With the default multi-backbone fallback, `accepted_name` is the pick of the highest-priority backbone that matched at the best quality reached. To see where backbones disagree, pass `mode = "wide"` (or `"agreement"`) for each backbone's `accepted_name` side by side; to follow one authority, name a single `backbone`.

For example, the red and parma kangaroos: the GBIF Backbone Taxonomy accepts `Macropus rufus` and `Macropus parma`, treating `Osphranter rufus` and `Notamacropus parma` as synonyms of them, so `taxify("Osphranter rufus", backbone = "gbif")` resolves to `Macropus rufus`. The Catalogue of Life splits the genus and does the reverse, so `taxify("Macropus rufus", backbone = "col")` resolves to `Osphranter rufus`. Both are faithful to their source; the difference is in the backbones, not in the matching.

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

# Match a few names
taxify(c("Quercus robur", "Pinus sylvestris"))

# Disable fuzzy matching
taxify("Quercus robus", fuzzy = FALSE)

# Constrain fuzzy candidates to a geographic region: a TDWG Level 3 code,
# or a region name resolved via the bundled WGSRPD crosswalk
taxify("Quercus robus", region = "BGM")
taxify("Quercus robus", region = "Belgium")

# Constrain by coordinates (downloads WGSRPD boundaries on first use)
## Not run:
taxify("Quercus robus", coords = c(4.35, 50.85))

## End(Not run)

# Fallback chain: try WFO first, then COL for unmatched
taxify(c("Quercus robur", "Panthera leo"),
       backbone = c("wfo", "col"))

# Compare how two backbones resolve the same names, side by side
taxify(c("Quercus robur", "Pinus sylvestris"),
       backbone = c("wfo", "col"), mode = "wide")

options(old)
```

taxify_build *Build a backbone database from source*

Description

Builds the backbone `.vtr` for a backend from its upstream Darwin Core source, delegating to the sibling `taxifydb` package (which must be installed). This is the from-source path, for rebuilding a backbone locally.

Usage

```
taxify_build(backend, dest = NULL, verbose = TRUE, ...)
```

Arguments

`backend`	A `taxify_backend` object or a character string (e.g., `"wfo"`).
`dest`	Character. Destination directory. Defaults to `taxify_data_dir()`.
`verbose`	Logical. Print progress messages.
`...`	Additional arguments passed to methods.

Details

Everyday use does not need this: `taxify()` auto-downloads the pre-built `.vtr` on first use, and `taxify_download()` fetches a pre-built backbone directly without `taxifydb`.

Value

The path to the `.vtr` file (invisibly).

See Also

`taxify_download()` to fetch a pre-built backbone, `taxify()` which downloads on first use.

taxify_build_register *Build the genus register from source*

Description

Rebuilds `genus_register.vtr` and `backend_coverage.vtr` locally instead of downloading the published pair. The build itself lives in `taxifydb`, which resolves every backbone in the register's fixed backbone set and can take well over an hour; the download takes seconds and yields the same file, so this is an escape hatch rather than the normal route.

Usage

```
taxify_build_register(verbose = TRUE)
```

Arguments

verbose	Logical. Print progress messages. Default TRUE.

Details

taxify_download("register") fetches the published pair.

Value

Path to genus_register.vtr (invisibly).

See Also

taxify_load_register() to load it into memory, taxify_register_coverage() to query backbone coverage for a genus.

---

taxify_clean *Remove downloaded taxify data*

Description

Frees disk space in the data directory (taxify_data_dir()). Called with no arguments it removes what taxify no longer reads: builds replaced by a newer one or kept by version, and files left by an interrupted download. The active build of every backbone and enrichment stays, unless unused_days is given. Name assets to remove them completely, or pass "all" to delete the data directory.

Usage

```
taxify_clean(
  assets = NULL,
  unused_days = NULL,
  dry_run = FALSE,
  verbose = TRUE
)
```

Arguments

assets	NULL (default) for superseded builds and interrupted downloads, a character vector of backbone or enrichment names to remove completely, or "all" for the whole data directory.
unused_days	Number or NULL. With assets = NULL, also remove every outdated build, the active one included, not read or downloaded for this many days.
dry_run	Logical. List what would be removed, without removing it.
verbose	Logical. Report the space freed.

Details

A removed backbone or enrichment downloads again when it is next installed; a lockfile pin is restored by `taxify_restore()` from the published copy of that build.

Value

Invisibly, a data.frame with one row per path removed (or, with `dry_run = TRUE`, that would be removed): `component`, `type` (`"backbone"`/`"enrichment"`/`"directory"`), `reason` (`"superseded"`, `"unused"`, `"interrupted"`, `"removed"`), `size_mb` and `path`.

Outdated builds left unused

Every time a session first reads a build, taxify records the time. When the package is loaded it removes from the data directory every outdated build that has not been read or downloaded for `getOption("taxify.max_unused_days")` days (default 90), and the files of interrupted downloads. A build is outdated when it is stored in the format used before taxify 0.7.0, or when a newer build of it has been published. The current build of a backbone or enrichment stays however long it is left unused, and so does a build pinned by `taxify_restore()`. Set the option to `Inf` to keep every build. `taxify_clean(unused_days = )` applies the same rule on demand to the data directory in use.

See Also

`taxify_store()` for the builds on disk, `install_backbones()`.

Examples

```
old <- options(taxify.data_dir = file.path(tempdir(), "taxify_clean_demo"))
taxify_clean(dry_run = TRUE)
taxify_clean(unused_days = 30, dry_run = TRUE)
options(old)
```

---

taxify_clear_cache *Clear all cached backbones*

Description

Removes all loaded backbone handles from memory. The next call to `taxify()` will re-load from disk.

Usage

```
taxify_clear_cache()
```

Value

No return value, called for side effects.

`taxify_databases` *One overview of every database taxify knows about*

Description

Stacks the taxonomic backbones (from `list_backbones()`) and the trait/status enrichments (from `list_enrichments()`) into a single frame with a `type` column, so a new user can see the full breadth in one call rather than needing to know three separate discovery verbs. The cross-source trait vocabulary is summarised in the message footer; browse it with `list_traits()`.

Usage

```
taxify_databases(verbose = TRUE)
```

Arguments

`verbose`	Logical. When `TRUE` (default), prints a one-line count of backbones, enrichments, and registered traits.

Value

A data.frame with columns: `type` (`"backbone"` or `"enrichment"`), `name`, `scope` (taxonomic scope for backbones; provided trait columns for enrichments), `n_rows`, `version`, `source_date` (the date of the upstream data, where recorded, which can be much earlier than the release `version` that packaged it), `installed`, `source`.

See Also

`list_backbones()`, `list_enrichments()`, `list_traits()`.

Examples

```
## Not run:
taxify_databases()

## End(Not run)
```

taxify_data_dir *Get the taxify data directory*

Description

Returns the directory where taxify stores downloaded backbone and enrichment `.vtr` files. By default this is the platform-appropriate per-user cache returned by `tools::R_user_dir()` (available since R 4.0).

Usage

```
taxify_data_dir()
```

Details

The location can be overridden, in order of precedence, by the `taxify.data_dir` option (`getOption("taxify.data_dir")`) or the `TAXIFY_DATA_DIR` environment variable. This is useful to point taxify at a shared cache, or at the small bundled example database returned by `taxify_example_data()`.

Value

Character string. Path to the data directory.

taxify_download *Download a pre-built taxify backbone*

Description

Downloads a pre-built `.vtr` backbone from GitHub Releases using the taxify manifest. This needs no build tools and does not require `taxifydb`; it is the fast path `taxify()` uses internally on first use. Call it directly to pre-fetch backbones before an offline session. Progress is always shown; no prompts are shown, so calling this function is consent.

Usage

```
taxify_download(
  backbone = "wfo",
  version = "latest",
  content_id = NULL,
  verbose = TRUE
)
```

Arguments

backbone	Character. A backbone name (e.g. "wfo", "col", "gbif"; list_backbones() lists them all) or "register" for the genus register. Multiple backbones can be given as a character vector.
version	Character. "latest" (default) downloads into <data_dir>/<backbone>/latest/ and will be overwritten on future updates. A specific version string (e.g., "2024.01") downloads into a pinned folder that is never overwritten.
content_id	Character or NULL (default). The content id of one exact build, the md5 of its .vtr, as recorded by taxify_lock() and by the manifest. Given one, taxify fetches that build from the immutable copy published beside the rolling asset, verifies the bytes hash back to the id, and makes it the active backbone; the build it replaces is kept under its own content id. Pass one id per backbone, or one shared by all of them.
verbose	Logical. Default TRUE.

Details

If no pre-built .vtr is available for a backbone, it falls back to building from source via taxify_build() (which requires taxifydb).

Value

The path(s) to the downloaded .vtr file(s) (invisibly).

See Also

taxify_build() to build a backbone from source via taxifydb, taxify_download_enrichment() for published datasets, taxify_store() for the builds already on disk.

---

taxify_download_enrichment

Download one or more enrichment .vtr files

Description

Downloads pre-built enrichment .vtr files from the taxify manifest.

Usage

```
taxify_download_enrichment(
  enrichment,
  version = "latest",
  content_id = NULL,
  verbose = TRUE
)
```

Arguments

`enrichment`	Character. One or more enrichment names (e.g., `"iucn"`, `"griis"`, `"zanne"`).
`version`	Character. `"latest"` (default) or a specific version string.
`content_id`	Character or `NULL` (default). The content id of one exact build, the md5 of its `.vtr`, as recorded by `taxify_lock()` and by the manifest. Given one, taxify fetches that build from the immutable copy published beside the rolling asset, verifies the bytes hash back to the id, and makes it the active build; the build it replaces is kept on disk under its own content id. This is what turns a lockfile's recorded id back into bytes. Pass one id per `enrichment`, or one shared by all of them.
`verbose`	Logical. Default `TRUE`.

Details

Available enrichments:

iucn IUCN conservation status (LC/NT/VU/EN/CR/EW/EX)

griis GRIIS invasive species status by country

zanne Zanne et al. 2014 woody/herbaceous classification

wcvp WCVP native range by TDWG botanical region

eive EIVE 1.0 ecological indicator values (European plants)

diaz_traits Diaz et al. 2022 seed mass and plant height

elton_traits EltonTraits 1.0 diet and foraging (birds + mammals)

avonet AVONET bird morphology and migration

pantheria PanTHERIA mammal life-history traits

common_names GBIF vernacular names (multi-language)

amphibio AmphiBIO amphibian life-history and ecological traits

leda LEDA Traitbase NW European plant traits (Kleyer et al. 2008)

Value

The path(s) to the downloaded `.vtr` file(s) (invisibly).

See Also

`taxify_store()` for the builds already on disk, `taxify_restore()` to reinstall everything a lockfile pins.

taxify_example_data *Path to the bundled example database*

Description

taxify ships a tiny example database (a handful of species per backbone plus matching enrichment tables) so that examples and quick experiments run offline, without downloading the full multi-million-row backbones.

Usage

```
taxify_example_data()
```

Details

Point taxify at it for the current session by setting the `taxify.data_dir` option:

```
old <- options(taxify.data_dir = taxify_example_data())
taxify("Quercus robur") |> add_zanne()
options(old)  # restore the real data directory
```

The example database is read-only and covers only the species used in the package examples; use the full downloaded backbones for real work.

Value

Character string. Path to the bundled example database directory, or `""` if it is not installed.

See Also

`taxify_data_dir()`

taxify_ids *List every accepted ID of each matched name*

Description

`taxify()` returns one accepted ID per name, and, when any name has several, records in `accepted_ids` every accepted taxon the backbone files the name under: a homonym published by two authors, or a name held twice, once accepted and once as a doubtful or duplicate record. This verb lays those out one row per (name, accepted ID), resolved against the backbone, with the ID `taxify()` picked marked. On the GBIF backbone each ID carries its occurrence count, so the IDs to request occurrence data with can be read off directly.

Usage

```
taxify_ids(x, verbose = TRUE)
```

Arguments

x	A taxify() result.
verbose	Logical. Default TRUE.

Value

A data.frame with one row per (matched input row, accepted ID), in input order and, within a name, in the order of accepted_ids (the pick first):

input_name The queried name.

backbone Backbone that matched the name.

accepted_id An accepted ID the name resolves to.

gbif_key The legacy GBIF taxon key(s) of that taxon, the keys GBIF serves occurrence records for: the ID itself on the gbif backbone, and on a backbone whose build carries a crosswalk (COL XR) the |-delimited set of GBIF keys for the same name, several when GBIF files it under more than one. NA on a backbone with no crosswalk.

accepted_name The name of that accepted taxon.

authorship Its authorship.

rank Its rank.

taxonomic_status Its status in the backbone ("ACCEPTED", GBIF's "DOUBTFUL", WFO's "UNCHECKED", ...).

family Its family.

n_occurrences GBIF occurrence records under this ID, including those of its synonyms and descendants (what a GBIF download by this key returns), as counted when the backbone was built. NA on a backbone without counts, and for an ID whose count was not taken.

is_pick Logical. Is this the accepted_id taxify() reported?

role What the ID is to the queried name: "pick" (the accepted_id taxify() reported), "same_name" (another record of the same taxon, or a record keeping the queried name as its own concept, as GBIF's doubtful homonym records of *Pinus sylvestris* do) or "other_taxon" (the accepted taxon of a synonym record under another author, a different species: GBIF files *Absinthium vulgare* Dulac under *Artemisia vulgaris* subsp. *vulgaris* while the pick, Lam., goes to *A. absinthium*).

pick_basis What decided the pick over the name's other accepted IDs, the same on every row of a name: "status", "priority" (publication year), "occurrences", "validity", "fuzzy", or "id" when only the lowest taxon ID separated them, which settles nothing about the taxon the name denotes; "authorship", "basionym" or "kingdom" when the input's author, the record's basionym link or the kingdom argument of taxify() moved the pick. NA for a name with one ID.

Unmatched names contribute no rows.

See Also

taxify(), synonyms().

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

taxify(c("Quercus robur", "Pinus sylvestris")) |>
  taxify_ids()

options(old)
```

---

taxify_load_register *Load the unified genus register into memory*

Description

Reads genus_register.vtr and caches it as a data.frame in .taxify_env$register, downloading it on first use. Subsequent calls reuse the cached version unless force = TRUE.

Usage

```
taxify_load_register(force = FALSE, verbose = TRUE)
```

Arguments

force	Logical. If TRUE, reloads from disk even if already cached. Default FALSE.
verbose	Logical. Print progress messages. Default TRUE.

Details

The register contains one row per genus with columns genus, kingdom, phylum, class, order, family, kingdom_group, taxon_group and life_form.

Value

The register data.frame (invisibly).

`taxify_lock` *Record the exact backbone and enrichment versions behind a result*

Description

Writes a machine-readable lockfile pinning which backbone and enrichment assets (name, version, and byte-identity, the content id) produced a `taxify()` result. Where `cite()` prints prose citations, `taxify_lock()` records the reproducible counterpart, so a manuscript's Methods can state exactly what was matched against and `taxify_restore()` can later verify an install still matches.

Usage

```
taxify_lock(x = NULL, file = NULL, verbose = TRUE)
```

Arguments

`x`	A `taxify()` result (its resolved backbones and any `add_*()` published datasets are locked), or `NULL` (default) to snapshot every installed backbone.
`file`	Optional path to write the lockfile to (JSON). When `NULL` (default) nothing is written and the lock is only returned.
`verbose`	Logical. Default `TRUE`.

Value

A lock list (invisibly when `file` is written), with elements `taxify_version`, `created`, `r_version`, `backbones`, and `enrichments`.

See Also

`taxify_restore()` to check an install against a lockfile, `cite()` for prose citations.

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

res  <- taxify("Quercus robur", backbone = "wfo", verbose = FALSE)
lock <- taxify_lock(res)
lock$backbones[[1]]$name

options(old)
```

taxify_long *Reshape grouped enrichment columns to long format*

Description

Converts wide-format columns produced by grouped enrichments (e.g., `invasive_status_AT`, `invasive_status_DE`) back to long format with one row per species x group combination.

Usage

```
taxify_long(x, cols = NULL, group_col = NULL, drop_na = FALSE)
```

Arguments

x	A data.frame, typically a `taxify()` result after applying a grouped enrichment like `add_griis()`, `add_alien_first_records()`, or `add_wcvp()`.
cols	Character vector of base column names to reshape. These are the column names without the group suffix (e.g., `"invasive_status"`, not `"invasive_status_AT"`). If omitted, auto-detected from the enrichment metadata stamped by the `add_*()` functions.
group_col	Character. Name for the output group column. If omitted, auto-detected from enrichment metadata (e.g., `"country_code"` for invasive status, `"location"` for alien first records).
drop_na	Logical. If `TRUE`, drop rows where all value columns are `NA`. Default `FALSE`.

Details

When `cols` and `group_col` are omitted, `taxify_long()` reads the reshape metadata attached by grouped enrichment functions (`add_griis()`, `add_alien_first_records()`, `add_wcvp()`, `add_common_names()`). If multiple grouped enrichments were applied, all are reshaped together (they must share the same group column).

Column matching uses the explicit base names in `cols` to avoid ambiguity. For example, given `cols = c("alien_first_record", "alien_first_record_source")`, the column `alien_first_record_source_AT` is correctly matched to base `alien_first_record_source` (not `alien_first_record` with suffix `source_AT`), because longer base names are matched first.

If the columns in `x` exactly match `cols` (no suffixed variants), the data is already in single-group format. In this case, the data.frame is returned unchanged with `group_col` set to `NA`.

Value

A data.frame in long format. All columns from `x` that are not part of the reshape are preserved. The reshaped columns use their base names (without suffix), and a new `group_col` column contains the group code extracted from the suffix.

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

# Auto-detected: no cols or group_col needed
taxify("Robinia pseudoacacia") |>
  add_alien_first_records(location = c("AT", "DE")) |>
  taxify_long()

# Explicit: override auto-detection
taxify("Robinia pseudoacacia") |>
  add_griis(country = c("AT", "DE")) |>
  taxify_long(cols = "invasive_status", group_col = "country")

options(old)
```

`taxify_pin` *Pin or release installed taxify assets*

Description

A pinned build stays active: the version check that `taxify()`, `install_backbones()` and the `add_<source>()` functions run against the manifest leaves it in place instead of replacing it with the current release. Pin the builds a project was run against to keep a shared data directory from moving under it, and release them when the project is done. The pin is recorded in the build's `meta.json` together with its content id (the md5 of its `.vtr`, as `taxify_lock()` records it), so a pinned build can also be named in a lockfile.

Usage

```
taxify_pin(name, pin = TRUE, kind = NULL, verbose = TRUE)
```

Arguments

`name`	Character vector of installed backbone and/or enrichment names.
`pin`	Logical. `TRUE` (default) pins; `FALSE` releases the pin, so the next version check compares the build against the manifest again.
`kind`	`NULL` (default) to find each name among the installed backbones and enrichments, or `"backbone"` / `"enrichment"` to look only there. Needed for a name installed as both (e.g. `"wcvp"`).
`verbose`	Logical. Default `TRUE`.

Details

Every name is checked before anything is written: when one is not installed, or is installed as both a backbone and an enrichment and `kind` does not say which, nothing is pinned.

`taxify_restore()` with `install = TRUE` pins every build matching a lockfile, and a build fetched with `taxify_download(content_id = )` or `taxify_download_enrichment(content_id = )` is pinned when it is activated.

Value

A data.frame with one row per name: `component`, `type`, `version`, `content_id` (the full id of the build pinned or released) and `pinned`.

See Also

`taxify_restore()` to pin the builds a lockfile records, `taxify_store()` to list the builds on disk and whether each is pinned.

Examples

```
## Not run:
taxify_pin(c("wfo", "col"))
taxify_pin("iucn", kind = "enrichment")
taxify_pin(c("wfo", "col"), pin = FALSE)

## End(Not run)
```

`taxify_refresh_manifest`

Invalidate the session manifest cache

Description

Forces the next `fetch_manifest()` call to re-fetch from the network. Useful after the maintainer updates the manifest between R sessions without restarting R.

Usage

```
taxify_refresh_manifest()
```

Value

No return value, called for side effects.

`taxify_regions` *List the regions accepted by* `region=`

Description

Returns every region code and name the `region` argument of `taxify()` resolves, across both vocabularies it accepts, optionally filtered by a search term matched (case- and accent-insensitively) against the code and all three level names.

Usage

```
taxify_regions(search = NULL, scheme = c("all", "wgsrpd", "meow"))
```

Arguments

`search`	Optional character string. If supplied, only regions whose code or name contains it are returned.
`scheme`	Which vocabulary to list: `"all"` (the default), `"wgsrpd"` for the botanical regions, or `"meow"` for the marine ones.

Details

Botanical regions (`scheme = "wgsrpd"`) come from the bundled WGSRPD (World Geographical Scheme for Recording Plant Distributions) Level 3 crosswalk, and are also the codes `add_wcvp()` uses. Marine regions (`scheme = "meow"`) are the Marine Ecoregions of the World, listed only when the `marine_distribution` asset is installed, since that is also when `region=` resolves them. The two schemes nest the same way, so they share the three name columns: a MEOW ecoregion is the `name`, its province the `level2_name`, and its realm the `level1_name`.

Value

A data.frame with columns `code`, `name`, `level2_name`, `level1_name`, and `scheme`, one row per region.

Examples

```
head(taxify_regions())

taxify_regions("belgium")
taxify_regions("Europe")
```

taxify_register_coverage

Show backbone coverage for a genus

Description

Queries `backend_coverage.vtr` to determine which backbones contain the given genus, along with the backbone version at time of indexing.

Usage

```
taxify_register_coverage(genus)
```

Arguments

`genus`	Character scalar. The genus name to query.

Value

A data.frame with columns `genus`, `backbone`, `version`, `date_added`. Returns a zero-row data.frame if the genus is not found in any backbone.

taxify_restore

Check an install against a lockfile, and optionally reinstall what drifted

Description

Reads a lockfile written by `taxify_lock()` and reports, for each backbone and enrichment it pins, whether the currently installed asset matches (by version and byte-identity, the content id), has drifted, or is missing.

Usage

```
taxify_restore(file, install = FALSE, verbose = TRUE)
```

Arguments

`file`	Path to a lockfile written by `taxify_lock()`, or the lock list it returned.
`install`	Logical. `FALSE` (default) verifies and reports only. `TRUE` downloads and activates the pinned build of every asset that does not already match.
`verbose`	Logical. Default `TRUE`.

Details

With `install = TRUE` it also fetches the exact build each row pins, from the immutable copy published beside the rolling asset, and makes it the active one: the recorded run is put back in place in a single call. The build each pinned build replaces is kept on disk under its own content id, so restoring a lockfile is reversible. Every build that matches the lock afterwards is pinned, including one that already matched and needed no download, so the next session's version check does not refresh it away from the lock. Release a pin with `taxify_pin()`.

A build published before taxifydb began uploading an immutable copy cannot be recovered (the re-cut replaced it in place), and its row keeps reporting drift after the install pass.

Value

A data.frame with one row per pinned asset, columns: `component`, `type` (`"backbone"`/`"enrichment"`), `locked_version`, `installed_version`, `locked_content_id` and `installed_content_id` (short), and `status` (`"ok"`, `"version_drift"`, `"content_drift"`, `"missing"`, or `"unverified"` when what the lock pinned cannot be compared against this install). With `install = TRUE` the statuses describe the install afterwards, a `restored` column records which rows were fetched, and a `pinned` column which rows are now pinned (every row whose status is `"ok"`).

See Also

`taxify_lock()`, `taxify_pin()` to pin or release an installed build without a lockfile, `taxify_store()` for the builds on disk, `taxify_download_enrichment()` to fetch a single build by content id.

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

res  <- taxify("Quercus robur", backbone = "wfo", verbose = FALSE)
lock <- taxify_lock(res)
taxify_restore(lock, verbose = FALSE)

options(old)
```

---

taxify_store *Builds of a taxify asset held on disk*

Description

Lists each backbone and enrichment build in the data directory: the active one under `latest/`, and any other build in a directory named for its content id (the md5 of its `.vtr`, the same string `taxify_lock()` records). By default a refresh replaces a build in place, so an asset has one build; the others come from restoring a pinned build or from the `taxify.keep_*_versions` options below. `taxify_clean()` removes them.

Usage

```
taxify_store(asset = NULL)
```

Arguments

asset	Character. One or more backbone or enrichment names to list. NULL (default) lists every asset in the data directory.

Details

Use this to see which pinned builds a lockfile could be restored to without a download, and how much disk each asset takes.

Value

A data.frame with one row per build on disk, columns: component, type ("backbone"/"enrichment"), slot (the store directory: "latest" for the active build, else its content id), active, version, content_id, pinned, size_mb, downloaded_at.

Options

taxify.keep_enrichment_versions and taxify.keep_backbone_versions (both default FALSE) keep the build a refresh replaces, under its content id. Restoring a pinned build archives the build it replaces either way.

See Also

taxify_lock() to record a build, taxify_restore() to check or reinstall one, taxify_download_enrichment() to fetch one by content id, taxify_clean() to remove builds.

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())
head(taxify_store())
options(old)
```

---

trait_info *Describe a trait's sources and units*

Description

Prints the kind, canonical unit, and (for categorical traits) the shared vocabulary of a trait, and returns a data.frame of the sources add_trait() draws from: one row per source, with its enrichment, source column, citation, and the harmonization note (unit conversion or vocabulary mapping).

Usage

```
trait_info(trait)
```

Arguments

trait	Character. A single trait name; see list_traits().

Value

A data.frame (invisibly-friendly) with columns source, enrichment, column, citation, note (unit or vocabulary harmonization), and caution (a method or definition difference from the reference source, or NA). The header line (label, kind, unit, default priority, vocabulary) is printed as a message.

See Also

add_trait(), list_traits()

Examples

```
trait_info("woodiness")
```

---

upstream *List the higher classification (ancestors) of a taxon*

Description

Returns the lineage above taxon (its genus, family, order, class, phylum, and kingdom, for whichever of those ranks the backbone stores) as a tidy frame with one row per ancestor rank. Where downstream() reaches down to descendants, upstream() reaches up to ancestors; where add_classification() attaches the ranks to an existing taxify() result, upstream() takes a bare name. A synonym or misspelling is resolved to its accepted taxon first, so the lineage returned is the accepted taxon's.

Usage

```
upstream(taxon, backbone = NULL, to = NULL, ..., verbose = TRUE)
```

Arguments

taxon	A single taxonomic name (a species, genus, or higher taxon; synonyms and typos are resolved first).
backbone	A single backbone name or a taxify_backend object. NULL (default) uses the highest-priority installed backbone; name one that stores the higher ranks (e.g. "col") for a full lineage.
to	Optional rank (or ranks) to restrict the output to, e.g. to = "family" answers "what family is this in?" with a single row. NULL (default) returns the whole lineage.

...	Matching arguments passed to taxify() when resolving taxon (e.g. fuzzy, fuzzy_threshold, kingdom, region). Must be named.
verbose	Logical. Default TRUE.

Value

A data.frame with one row per ancestor rank, columns: input_name (the name as supplied), accepted_name (what it resolved to), rank, name, backbone, ordered kingdom -> genus. Empty when taxon does not resolve or the backbone stores no ranks above it.

See Also

downstream() for descendants, add_classification() to attach the ranks to a taxify() result, lowest_common().

Examples

```
# Runs offline against the bundled example database.
old <- options(taxify.data_dir = taxify_example_data())

# The lineage above a species (reptiledb carries the full higher hierarchy)
upstream("Naja naja", backbone = "reptiledb")

# Just the family
upstream("Naja naja", backbone = "reptiledb", to = "family")

options(old)
```

Index

add_algae_traits, 5
add_alien_first_records, 7
add_alien_first_records(), *156*, *194*
add_alpine_carabids, 8
add_amniote, 9
add_amphibio, 11
add_anage, 12
add_anage(), *46*
add_animaltraits, 13
add_arctic_traits, 14
add_arthropod_traits, 15
add_austraits, 16
add_avonet, 18
add_avonet(), *30*
add_bacdive, 19
add_baseflor, 20
add_baseflor(), *55, 56*, *73, 74*
add_bee_ostwald, 21
add_bet, 22
add_betsi_collembola_body_length, 24
add_betsi_collembola_traits, 25
add_betsi_earthworm_traits, 26
add_beukhof, 27
add_bien, 28
add_birdbase, 30
add_blanchard, 31
add_brot, 32
add_bryoatt, 33
add_ccdb, 34
add_cefas_btrait, 35
add_chelonians, 36
add_chowdhury, 38
add_chowdhury(), *9*, *54*, *69*
add_classification, 39
add_classification(), *152*, *163*, *171*, *201, 202*
add_clopla, 40
add_col_info, 41
add_col_info(), *39*
add_combine, 42
add_combine(), *44*, *46*, *119*
add_combine_imputed, 43
add_combine_imputed(), *43*, *45, 46*
add_combine_reported, 45
add_combine_reported(), *43, 44*
add_common_names, 46
add_common_names(), *152, 153*, *156*, *174, 175*, *194*
add_copepod_traits, 47
add_coral_traits, 48
add_data, 50
add_data(), *157*
add_diaz_traits, 52
add_diaz_traits(), *140*
add_disperse, 53
add_eberswalde, 54
add_eberswalde(), *9*
add_ecoflora, 55
add_ecoflora(), *73, 74*, *122*
add_ecomorphosis, 57
add_edwards_phyto, 58
add_eive, 59
add_eive(), *20*, *56*, *74*, *120*, *122*
add_ellers_collembola, 60
add_elton_traits, 61
add_epa_freshwater, 63
add_eupolltrait, 64
add_eurobat, 65
add_euromed_distribution, 66
add_euromed_distribution(), *157*
add_faprotax, 67
add_finand, 68
add_finand(), *99*
add_fishbase, 69
add_fishbase(), *71*, *130*
add_fishmorph, 71
add_fishtraits, 72
add_floraweb, 73

add_floraweb(), *55, 56*, *122*, *155*
add_freshwater_insects_conus, 75
add_frugivoria, 76
add_fungal_traits, 78
add_fungalroot, 77
add_fungalroot(), *112*
add_funguild, 79
add_gbif_info, 81
add_gbif_info(), *39*, *147*
add_gidias, 82
add_gift, 84
add_gift(), *140*, *155*, *162*
add_globi, 85
add_globi(), *144*
add_globtherm, 86
add_glonaf, 87
add_glonaf(), *90*
add_gmpd, 88
add_griis, 90
add_griis(), *135*, *156*, *194*
add_groot, 91
add_gwdd, 92
add_homerange, 93
add_hosts, 94
add_huang_amph, 95
add_hybrid_info, 96
add_hybrid_info(), *180, 181*
add_hydraulics, 97
add_imageomics_neon, 98
add_inrae_collembola_traits, 99
add_invacost, 100
add_invacost(), *144*
add_italic, 101
add_iucn, 103
add_kew_cvalues, 104
add_kew_cvalues(), *34, 35*
add_kew_sid, 105
add_leda, 106
add_leda(), *20*, *140*
add_leptraits, 107
add_madin, 109
add_monograph_collembola_body_length, 110
add_nesttrait, 111
add_noddb, 112
add_nztd, 113
add_octocoral, 114
add_odonata, 115
add_pantheria, 116
add_pantheria(), *46*, *119*
add_parravicini, 117
add_pelagic, 118
add_phylacine, 119
add_pignatti, 120
add_plantatt, 121
add_plantatt(), *33*
add_plazi_collembola_body_length, 122
add_pottier, 124
add_quimbayo, 125
add_ramond, 126
add_repttraits, 127
add_rimet_phyto, 128
add_saproxylic, 129
add_sealifebase, 130
add_sharkipedia, 132
add_sheld, 133
add_spider_traits, 134
add_sworm, 135
add_tetradensity, 136
add_thermofresh, 137
add_trait, 138
add_trait(), *23–25*, *27*, *55*, *58*, *60*, *82*, *97*, *100*, *103*, *110*, *123*, *147, 148*, *151*, *169*, *200, 201*
add_tree_of_sex, 141
add_usda_fungus_host, 142
add_useful_plants, 143
add_virion, 144
add_wcvp, 145
add_wcvp(), *67*, *156*, *194*, *197*
add_wfo_info, 146
add_zanne, 147
add_zanne(), *140*
add_zooplankton, 148

base::summary(), *146*

children, 149
children(), *154, 155*, *176*
cite, *68*, 150
cite(), *106*, *139*, *159–161*, *193*
class2tree, 151
class2tree(), *171*
comm2sci, 152
comm2sci(), *174, 175*

downstream, 154

downstream(), *149, 150*, *201, 202*

enrichment_cols, *6*, *8*, *10–15*, *17*, *19, 20*, *22–28*, *30–35*, *37, 38*, *40*, *43–45*, *47*, *49*, *52–54*, *56–61*, *63–65*, *67, 68*, *70–72*, *75, 76*, *78*, *80*, *82*, *85, 86*, *89*, *91–95*, *97, 98*, *100–106*, *108–119*, *121*, *123–127*, *129, 130*, *132–137*, *141, 142*, *144*, *147, 148*, 155
enrichment_cols(), *18*, *46*, *66*, *74*, *90*, *143*, *145*, *156*, *162*
enrichment_groups, 156
euromed_areas, 157
euromed_areas(), *66, 67*
export_data, 157

gbif_backmatch, 158
gbif_backmatch(), *159, 160*
gbif_fetch, 159
gbif_request, 160
gbif_request(), *158–160*
gift_traits, 162
gift_traits(), *84, 85*

id2name, 163
id2name(), *96*
inspect, 164
install_backbones, 166
install_backbones(), *179*, *185*, *195*

list_backbones, 167
list_backbones(), *167*, *186*, *188*
list_enrichments, 168
list_enrichments(), *155, 156*, *167, 168*, *186*
list_traits, 169
list_traits(), *138*, *140*, *168*, *186*, *201*
lookup_genus, 170
lowest_common, 170
lowest_common(), *152*, *202*

parse_name, 171
print.taxify_tree *(*class2tree*)*, 151

reconcile, 173

sci2comm, 174
synonyms, 176
synonyms(), *150*, *155*, *163*, *174*, *192*

taxify, 177
taxify(), *5–28*, *30–41*, *43–50*, *52–61*, *63–82*, *84–130*, *132–138*, *141–148*, *150*, *152–155*, *157, 158*, *160, 161*, *163, 164*, *166, 167*, *170–176*, *183*, *185*, *187*, *190–195*, *197*, *201, 202*
taxify_build, 183
taxify_build(), *188*
taxify_build_register, 183
taxify_clean, 184
taxify_clean(), *166*, *199, 200*
taxify_clear_cache, 185
taxify_data_dir, 187
taxify_data_dir(), *166*, *183, 184*, *190*
taxify_databases, 186
taxify_databases(), *168*
taxify_download, 187
taxify_download(), *183*
taxify_download_enrichment, 188
taxify_download_enrichment(), *188*, *199, 200*
taxify_example_data, 190
taxify_example_data(), *187*
taxify_ids, 190
taxify_ids(), *158–161*, *174*, *181*
taxify_load_register, 192
taxify_load_register(), *184*
taxify_lock, 193
taxify_lock(), *188, 189*, *195*, *198–200*
taxify_long, 194
taxify_long(), *51*
taxify_pin, 195
taxify_pin(), *167*, *199*
taxify_refresh_manifest, 196
taxify_regions, 197
taxify_regions(), *145*, *166*, *178*
taxify_register_coverage, 198
taxify_register_coverage(), *184*
taxify_restore, 198
taxify_restore(), *167*, *185*, *189*, *193*, *196*, *200*
taxify_store, 199
taxify_store(), *185*, *188, 189*, *196*, *199*
tools::R_user_dir(), *187*
trait_info, 200
trait_info(), *138–140*, *169*

upstream, 201
upstream(), *155*