Package: taxifydb (via r-universe)

June 19, 2026

Title Build Backbones and Enrichments for the 'taxify' Package

Version 0.1.0

Description Build pipeline for the 'taxify' package. Downloads raw source data from official providers (WFO, COL, GBIF, ITIS, NCBI Taxonomy, Open Tree of Life, WoRMS, Euro+Med PlantBase, Index Fungorum, AlgaeBase) and a wide set of trait and conservation datasets, normalizes to a unified Darwin Core-like schema, and writes pre-compiled '.vtr' files that the 'taxify' runtime consumes. Separates build-time concerns (network access, parsing, schema normalization) from runtime concerns so that 'taxify' itself stays lean.

License MIT + file LICENSE

URL https://github.com/gcol33/taxifydb

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

Depends R (>= 4.1.0)

Imports curl, digest, jsonlite, taxify, utils, vectra

Suggests DBI, openxlsx2, rfishbase, RSQLite, testthat (>= 3.0.0)

Remotes gcol33/taxify

Config/testthat/edition 3

Encoding UTF-8

Roxygen list(markdown = TRUE)

Config/roxygen2/version 8.0.0

Config/pak/sysreqs make libssl-dev

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

Date/Publication 2026-06-19 21:26:26 UTC

RemoteUrl https://github.com/gcol33/taxifydb

RemoteRef HEAD

RemoteSha 76810a268e76024e760fc2d9de53e60993800e48

Contents

apply_delta *Apply a binary delta to produce a new .vtr*

Description

Apply a binary delta to produce a new .vtr

Usage

```
apply_delta(old_path, delta_path, new_path)
```

Arguments

old_path	Character. Path to the current local .vtr.
delta_path	Character. Path to the downloaded .xdelta file.
new_path	Character. Output path for the patched .vtr.

Value

The new path (invisibly), or NULL on failure.

build_algaebase *Build the AlgaeBase backbone .vtr*

Description

Build the AlgaeBase backbone .vtr

Usage

```
build_algaebase(
  output_dir = "output/algaebase",
  version = NULL,
  verbose = TRUE
)
```

Arguments

output_dir	Character.
version	Character or NULL.
verbose	Logical.

Value

Path to the .vtr file (invisibly).

build_all_name_lookups

Build name-lookup tables for all installed taxify backbones

Description

Locates each backbone in the user's taxify data dir and writes a `{backend}_name_lookup.vtr` alongside it.

Usage

```
build_all_name_lookups(
  backends = c("wfo", "col", "gbif", "itis", "ncbi", "ott", "worms"),
  overwrite = FALSE
)
```

Arguments

`backends`	Character vector. Default: 7 standard backends.
`overwrite`	Logical. Rebuild even if the lookup .vtr already exists.

Value

Character vector of paths to the built lookups.

build_backend

Build a taxify backbone `.vtr` *file from source*

Description

Single entry point for backbone builds. Downloads the raw source, normalizes to the unified schema, precomputes matching keys, and writes the final `.vtr` with indexes and metadata sidecar.

Usage

```
build_backend(name, output_dir = NULL, version = NULL, verbose = TRUE)
```

Arguments

`name`	Character. Backend identifier (e.g., "itis"). See `list_backends()` for available names.
`output_dir`	Character. Output directory. Default: `output/<name>`.
`version`	Character or NULL. Version string for the build. If `NULL`, defaults to the current YYYY.MM (or the backend's bundled default).
`verbose`	Logical.

Value

Path to the built `.vtr` file (invisibly).

---

build_col *Build the COL backbone .vtr*

Description

Also writes the SpeciesProfile.tsv (extinct/marine flags) as a separate `col_species_profile.vtr` next to the main backbone, if present.

Usage

```
build_col(output_dir = "output/col", version = NULL, verbose = TRUE)
```

Arguments

`output_dir`	Character.
`version`	Character or NULL.
`verbose`	Logical.

Value

Path to the .vtr file (invisibly).

---

build_enrichment *Build an enrichment* `.vtr` *from source*

Description

Downloads the raw source for an enrichment, parses it via the registered parser, expands names across all 7 taxify backbones via `resolve_enrichment_names()`, and writes a `.vtr` plus `meta.json` sidecar via `build_enrichment_vtr()`.

Usage

```
build_enrichment(
  name,
  output_dir = NULL,
  version = NULL,
  url = NULL,
  resolve_names = TRUE,
  verbose = TRUE
)
```

Arguments

name	Character. Enrichment identifier. See list_enrichments().
output_dir	Character or NULL. Output directory. Default output/enrichment/<name>.
version	Character or NULL. Version override. If NULL, uses the registry's pinned version.
url	Character or NULL. Custom source URL override. If supplied, the build version defaults to format(Sys.Date(), "%Y.%m") and source_doi is set to NULL.
resolve_names	Logical. Run resolve_enrichment_names() before writing. Default TRUE.
verbose	Logical. Default TRUE.

Value

Path to the built .vtr file (invisibly).

---

build_enrichment_vtr *Write an enrichment .vtr file*

Description

Sorts by canonical_name, writes the .vtr, creates hash indexes on canonical_name (and optionally a group column), and writes a meta.json sidecar.

Usage

```
build_enrichment_vtr(
  df,
  vtr_path,
  name,
  version,
  source_url,
  source_doi = NULL,
  license = "unknown",
  attribution = NULL,
  group_col = NULL,
  batch_size = 50000L
)
```

Arguments

df	A data.frame with at least a canonical_name column.
vtr_path	Character. Output path for the .vtr file.
name	Character. Enrichment identifier (e.g., "woodiness").
version	Character. Version string (e.g., "2026.04").
source_url	Character. URL the source data was downloaded from.
source_doi	Character or NULL. DOI of the source dataset.

license	Character. License string (e.g., "CC0", "CC BY 4.0").
attribution	Character. Human-readable attribution string.
group_col	Character or NULL. Column to index for group-based enrichments (e.g., "country_code", "tdwg_code", "lang").
batch_size	Integer. Row group size for vectra (default 50000).

Value

The path to the .vtr file (invisibly).

`build_euromed` *Build the Euro+Med backbone .vtr*

Description

Build the Euro+Med backbone .vtr

Usage

```
build_euromed(output_dir = "output/euromed", version = NULL, verbose = TRUE)
```

Arguments

output_dir	Character.
version	Character or NULL.
verbose	Logical.

Value

Path to the .vtr file (invisibly).

`build_fungorum` *Build the Species Fungorum Plus backbone .vtr*

Description

Build the Species Fungorum Plus backbone .vtr

Usage

```
build_fungorum(output_dir = "output/fungorum", version = NULL, verbose = TRUE)
```

Arguments

output_dir	Character.
version	Character or NULL.
verbose	Logical.

Value

Path to the .vtr file (invisibly).

---

build_gbif *Build the GBIF backbone .vtr*

Description

Build the GBIF backbone .vtr

Usage

```
build_gbif(output_dir = "output/gbif", version = NULL, verbose = TRUE)
```

Arguments

output_dir	Character.
version	Character or NULL.
verbose	Logical.

Value

Path to the .vtr file (invisibly).

---

build_itis *Build the ITIS backbone .vtr from scratch*

Description

Downloads the ITIS SQLite dump, normalizes it, precomputes matching keys, embeds synonym info, and writes the final .vtr with indexes.

Usage

```
build_itis(output_dir = "output/itis", version = NULL, verbose = TRUE)
```

Arguments

output_dir	Character. Directory for the output .vtr.
version	Character or NULL. If NULL, uses YYYY.MM of build date.
verbose	Logical.

Value

Path to the .vtr file (invisibly).

---

build_name_lookup *Build a name-lookup .vtr from a backbone .vtr*

Description

Build a name-lookup .vtr from a backbone .vtr

Usage

```
build_name_lookup(bb_path, out_path, verbose = TRUE)
```

Arguments

bb_path	Character. Backbone .vtr path.
out_path	Character. Lookup .vtr destination.
verbose	Logical.

Value

out_path (invisibly).

---

build_ncbi *Build the NCBI backbone .vtr*

Description

Build the NCBI backbone .vtr

Usage

```
build_ncbi(output_dir = "output/ncbi", version = NULL, verbose = TRUE)
```

Arguments

output_dir	Character.
version	Character or NULL.
verbose	Logical.

Value

Path to the .vtr file (invisibly).

---

build_ott *Build the OTT backbone .vtr*

Description

Build the OTT backbone .vtr

Usage

```
build_ott(output_dir = "output/ott", version = NULL, verbose = TRUE)
```

Arguments

output_dir	Character.
version	Character or NULL.
verbose	Logical.

Value

Path to the .vtr file (invisibly).

---

build_vtr *Write a backbone .vtr file from a precomputed data.frame*

Description

Sorts by genus (for zone-map pruning), writes the .vtr, creates hash indexes, and writes a metadata sidecar.

Usage

```
build_vtr(df, vtr_path, backend_name, version, source_url, batch_size = 50000L)
```

Arguments

df	A precomputed backbone data.frame (from precompute_backbone()).
vtr_path	Character. Output path for the .vtr file.
backend_name	Character. Backend identifier (e.g., "itis").
version	Character. Version string.
source_url	Character. URL the source data was downloaded from.
batch_size	Integer. Row group size for vectra (default 50000).

Value

The path to the .vtr file (invisibly).

---

build_wfo *Build the WFO backbone .vtr from source*

Description

Build the WFO backbone .vtr from source

Usage

```
build_wfo(output_dir = "output/wfo", version = NULL, verbose = TRUE)
```

Arguments

output_dir	Character. Output directory.
version	Character or NULL. Defaults to the bundled WFO release tag.
verbose	Logical.

Value

Path to the .vtr file (invisibly).

build_worms *Build the WoRMS backbone .vtr*

Description

Also writes the SpeciesProfile.tsv (habitat flags) as a separate `worms_species_profile.vtr` next to the main backbone.

Usage

```
build_worms(output_dir = "output/worms", version = NULL, verbose = TRUE)
```

Arguments

`output_dir`	Character.
`version`	Character or NULL.
`verbose`	Logical.

Value

Path to the .vtr file (invisibly).

check_all_enrichment_versions *Check all non-static enrichments in a manifest for version freshness*

Description

Check all non-static enrichments in a manifest for version freshness

Usage

```
check_all_enrichment_versions(manifest_path = "manifest/manifest.json")
```

Arguments

`manifest_path`	Character. Path to manifest.json.

Value

Data.frame with columns: name, source_version, upstream_version, outdated, check_url, note.

check_dryad_version *Check a Dryad dataset for the latest version*

Description

Check a Dryad dataset for the latest version

Usage

```
check_dryad_version(source_url)
```

Arguments

source_url	Character. Dryad download URL containing a DOI.

Value

Named list with `version` and `url`.

check_enrichment_source_version *Dispatch version check based on a manifest entry's source format*

Description

Dispatch version check based on a manifest entry's source format

Usage

```
check_enrichment_source_version(entry)
```

Arguments

entry	List. Manifest enrichment entry with `source_url`, `source_format`, `source_version`, `static`.

Value

Named list with `source_version`, `upstream_version`, `outdated`, `check_url`, and optionally `note`.

check_figshare_version

Check a Figshare article for the latest version

Description

Check a Figshare article for the latest version

Usage

```
check_figshare_version(source_url)
```

Arguments

source_url	Character. Figshare download URL.

Value

Named list with `version` and `url`.

check_gbif_api_version

Check the GBIF backbone API for the latest update date

Description

Check the GBIF backbone API for the latest update date

Usage

```
check_gbif_api_version(source_url)
```

Arguments

source_url	Character. (Unused; the GBIF backbone dataset ID is fixed.)

Value

Named list with `version` and `url`.

check_gbif_version *Check a GBIF hosted dataset for the Last-Modified date*

Description

Check a GBIF hosted dataset for the Last-Modified date

Usage

```
check_gbif_version(source_url)
```

Arguments

source_url	Character. GBIF hosted dataset URL.

Value

Named list with `version` (date string) and `url`.

check_wcvp_version *Check Kew WCVP for the latest version*

Description

WCVP has no dedicated API, so we fall back to a HEAD request via `check_gbif_version()`.

Usage

```
check_wcvp_version(source_url)
```

Arguments

source_url	Character. WCVP download URL.

Value

Named list with `version` and `url`.

check_zenodo_version *Check a Zenodo record for the latest version*

Description

Check a Zenodo record for the latest version

Usage

```
check_zenodo_version(source_url)
```

Arguments

source_url	Character. Zenodo download URL.

Value

Named list with `version` (publication date) and `url`.

count_vtr_rows *Count rows in a .vtr file*

Description

Count rows in a .vtr file

Usage

```
count_vtr_rows(vtr_path)
```

Arguments

vtr_path	Character.

Value

Integer.

create_delta *Create a binary diff between two .vtr files*

Description

Create a binary diff between two .vtr files

Usage

```
create_delta(old_path, new_path, delta_path)
```

Arguments

old_path	Character. Path to the previous-version .vtr.
new_path	Character. Path to the new-version .vtr.
delta_path	Character. Output path for the .xdelta file.

Value

The delta path (invisibly), or NULL if xdelta3 is unavailable.

download_algaebase *Fetch all AlgaeBase records via /nameusage/search*

Description

Fetch all AlgaeBase records via /nameusage/search

Usage

```
download_algaebase(verbose = TRUE)
```

Arguments

verbose	Logical.

Value

A list of record objects (raw JSON list-of-lists).

download_and_unzip *Download a ZIP and extract, returning the path to a matching file*

Description

Downloads `url` to `dest_dir/source.zip` (if not cached), extracts into `dest_dir/extracted/`, and returns the first file matching `pattern`. If `pattern` is `NULL`, returns the extraction directory itself.

Usage

```
download_and_unzip(url, dest_dir, pattern = NULL)
```

Arguments

`url`	Character. URL to a ZIP archive.
`dest_dir`	Character. Directory for download and extraction.
`pattern`	Character or `NULL`. Regex to match the target file inside the ZIP. If `NULL`, returns the extraction directory itself.

Value

Path to the matched file, or the extraction directory if `pattern` is `NULL`.

download_col *Download and extract the COL DwC-A*

Description

Download and extract the COL DwC-A

Usage

```
download_col(dest = tempdir(), verbose = TRUE)
```

Arguments

`dest`	Character. Destination directory.
`verbose`	Logical.

Value

Path to the extracted COL directory.

download_curl_file *Download a file via curl*

Description

Downloads `url` into `dest_dir/filename` if the destination does not already exist (or is empty). Follows redirects, sets a User-Agent header, and optionally a Referer.

Usage

```
download_curl_file(url, dest_dir, filename, referer = NULL, user_agent = NULL)
```

Arguments

`url`	Character. URL to download.
`dest_dir`	Character. Directory to save into. Created if missing.
`filename`	Character. Output filename.
`referer`	Character or `NULL`. Optional Referer header.
`user_agent`	Character or `NULL`. Override the default User-Agent.

Value

Path to the downloaded file.

download_euromed *Download and extract the Euro+Med CSV*

Description

Download and extract the Euro+Med CSV

Usage

```
download_euromed(dest = tempdir(), verbose = TRUE)
```

Arguments

`dest`	Character. Destination directory.
`verbose`	Logical.

Value

Path to the extracted CSV.

download_fungorum *Download and extract the Species Fungorum Plus ColDP archive*

Description

Download and extract the Species Fungorum Plus ColDP archive

Usage

```
download_fungorum(dest = tempdir(), verbose = TRUE)
```

Arguments

dest	Character. Destination directory.
verbose	Logical.

Value

Path to the directory containing the extracted ColDP files.

download_gbif *Download the GBIF backbone simple.txt.gz*

Description

Download the GBIF backbone simple.txt.gz

Usage

```
download_gbif(dest = tempdir(), verbose = TRUE)
```

Arguments

dest	Character. Destination directory.
verbose	Logical.

Value

Path to the .gz file.

download_gbif_api_pages *Fetch paginated GBIF API results*

Description

Pages through a GBIF API endpoint that uses `offset` + `limit` and an `endOfRecords` flag. Returns a combined data.frame of `$results` payloads.

Usage

```
download_gbif_api_pages(base_url, params, limit = 1000L, max_pages = 100L)
```

Arguments

`base_url`	Character. GBIF API endpoint.
`params`	Named list. Query parameters (excluding `offset`/`limit`).
`limit`	Integer. Page size.
`max_pages`	Integer. Maximum pages to fetch.

Value

A data.frame of combined results. Empty data.frame if the endpoint returned no rows.

download_itis *Download and extract the ITIS SQLite database*

Description

Download and extract the ITIS SQLite database

Usage

```
download_itis(dest = tempdir(), verbose = TRUE)
```

Arguments

`dest`	Character. Destination directory.
`verbose`	Logical.

Value

Path to the extracted SQLite database file.

download_ncbi *Download and extract NCBI taxdump*

Description

Download and extract NCBI taxdump

Usage

```
download_ncbi(dest = tempdir(), verbose = TRUE)
```

Arguments

dest	Character. Destination directory.
verbose	Logical.

Value

Path to the extraction directory.

download_ott *Download and extract OTT taxonomy*

Description

Download and extract OTT taxonomy

Usage

```
download_ott(dest = tempdir(), verbose = TRUE)
```

Arguments

dest	Character. Destination directory.
verbose	Logical.

Value

Path to the extraction directory containing taxonomy.tsv.

download_wfo *Download and extract the WFO classification file*

Description

Download and extract the WFO classification file

Usage

```
download_wfo(dest = tempdir(), verbose = TRUE)
```

Arguments

dest	Character. Destination directory.
verbose	Logical.

Value

Path to the extracted classification file.

download_worms *Download and extract WoRMS DwC-A*

Description

Download and extract WoRMS DwC-A

Usage

```
download_worms(dest = tempdir(), verbose = TRUE)
```

Arguments

dest	Character. Destination directory.
verbose	Logical.

Value

Path to the extracted DwC-A directory.

enrichment_emergency_fallback *Build enrichment from source and return the raw data.frame*

Description

Same as `build_enrichment()` but returns the parsed data.frame instead of writing a `.vtr` file. Useful when the runtime needs the raw data in memory (e.g. emergency fallback wired into `taxify`).

Usage

```
enrichment_emergency_fallback(name, verbose = TRUE)
```

Arguments

`name`	Character. Enrichment identifier.
`verbose`	Logical. Default `TRUE`.

Value

data.frame with `canonical_name` + trait columns.

has_xdelta3 *Check if xdelta3 is available on PATH*

Description

Check if xdelta3 is available on PATH

Usage

```
has_xdelta3()
```

Value

Logical.

list_backends *List available backend builders*

Description

List available backend builders

Usage

```
list_backends()
```

Value

Character vector of backend identifiers.

list_enrichments *List available enrichment builders*

Description

List available enrichment builders

Usage

```
list_enrichments()
```

Value

Character vector of enrichment identifiers.

normalize_backbone *Normalize a raw backbone data.frame to the unified schema*

Description

Renames source-specific columns to canonical names and ensures consistent types and formatting (uppercase ranks/statuses, trimmed whitespace, etc.).

Usage

```
normalize_backbone(df, col_map, extra_cols = NULL)
```

Arguments

df	The raw data.frame from a backend's download/parse step.
col_map	A named list mapping canonical names to source column names: taxon_id, canonical_name, taxon_rank, taxonomic_status, accepted_name_usage_id, family, genus, specific_epithet, authorship, infraspecific_epithet.
extra_cols	Optional named list of additional columns to keep (canonical_name = source_name).

Value

A data.frame with standardized column names and formatting.

---

parse_algae_traits *Parse AlgaeTraits macroalgal traits (WoRMS ZIP export)*

Description

Parse AlgaeTraits macroalgal traits (WoRMS ZIP export)

Usage

```
parse_algae_traits(path)
```

Arguments

path	Character. Directory holding CSV/TXT/XLSX files extracted from the AlgaeTraits archive.

Value

data.frame with canonical_name + algae trait columns.

---

parse_alien_first_records *Parse Seebens et al. Global Alien Species First Record Database*

Description

Reads the "FirstRecords" sheet from the Seebens Excel file, maps region names to ISO 3166-1 alpha-2 codes, and deduplicates per species x country (keeping the earliest year).

Usage

```
parse_alien_first_records(path)
```

Arguments

path	Character. Path to the Seebens XLSX file.

Value

data.frame with canonical_name + country_code + first-record cols.

---

parse_amphibio *Parse AmphiBIO amphibian traits (CSV from ZIP)*

Description

Parse AmphiBIO amphibian traits (CSV from ZIP)

Usage

```
parse_amphibio(path)
```

Arguments

path	Character. Path to the AmphiBIO CSV.

Value

data.frame with canonical_name + trait columns.

---

parse_anage *Parse AnAge longevity and life-history traits (TSV from ZIP)*

Description

Parse AnAge longevity and life-history traits (TSV from ZIP)

Usage

```
parse_anage(path)
```

Arguments

path	Character. Path to anage_*.txt.

Value

data.frame with canonical_name + AnAge trait columns.

parse_animaltraits *Parse AnimalTraits observations CSV (aggregate to species medians)*

Description

Parse AnimalTraits observations CSV (aggregate to species medians)

Usage

```
parse_animaltraits(path)
```

Arguments

path	Character. Path to observations.csv.

Value

data.frame with canonical_name + body_mass_kg + metabolic_rate_w.

parse_arthropod_traits *Parse NW European Arthropod DwC-A (taxon + measurement + description)*

Description

Parse NW European Arthropod DwC-A (taxon + measurement + description)

Usage

```
parse_arthropod_traits(dir_path)
```

Arguments

dir_path	Character. Directory holding the DwC archive.

Value

data.frame with canonical_name + arthropod trait columns.

parse_avonet *Parse AVONET bird morphology XLSX*

Description

Parse AVONET bird morphology XLSX

Usage

```
parse_avonet(path)
```

Arguments

path	Character. Path to AVONET_BirdLife.xlsx.

Value

data.frame with canonical_name + morphology columns.

parse_common_names *Parse common names from GBIF, NCBI, and OTT*

Description

Merges vernacular names from three sources:

- GBIF: VernacularName.tsv (has ISO 639-1 language codes)
- NCBI: names.dmp where name_class == "common name" (no language)
- OTT: synonyms.tsv where type == "common name" (no language)

Usage

```
parse_common_names(dir_path)
```

Arguments

dir_path	Character. Directory containing gbif/, ncbi/, ott/ subdirs.

Details

NCBI and OTT common names have no language tag, so lang is set to NA.

Value

data.frame with canonical_name + lang + common_name.

parse_conservation_status *Parse IUCN Red List conservation status from the GBIF Darwin Core Archive*

Description

Reads the IUCN Red List archive published on GBIF (taxon.txt core plus the Distribution extension that carries `threatStatus`) and returns one global Red List category per accepted species-rank name. Reading the archive directly avoids the GBIF `species/search` endpoint, whose threat facet spans every checklist (so a name picks up conflicting categories from regional or erroneous lists) and whose offset ceiling truncates the large categories.

Usage

```
parse_conservation_status(dir_path)
```

Arguments

`dir_path`	Character. Directory holding the extracted IUCN archive (`taxon.txt`, `distribution.txt`).

Value

data.frame with canonical_name + conservation_status.

parse_diaz_traits *Parse Diaz et al. 2022 supplementary traits (XLSX)*

Description

Parse Diaz et al. 2022 supplementary traits (XLSX)

Usage

```
parse_diaz_traits(path)
```

Arguments

`path`	Character. Path to the Diaz 2022 XLSX.

Value

data.frame with canonical_name + seed_mass_mg + plant_height_m.

parse_eive *Parse EIVE 1.0 ecological indicator values (XLSX)*

Description

Parse EIVE 1.0 ecological indicator values (XLSX)

Usage

```
parse_eive(path)
```

Arguments

path	Character. Path to EIVE_1.0.xlsx.

Value

data.frame with canonical_name + indicator columns.

parse_elton_traits *Parse EltonTraits 1.0 birds + mammals TSVs*

Description

Parse EltonTraits 1.0 birds + mammals TSVs

Usage

```
parse_elton_traits(birds_path, mammals_path)
```

Arguments

birds_path	Character. Path to BirdFuncDat.txt.
mammals_path	Character. Path to MamFuncDat.txt.

Value

data.frame with canonical_name + diet/foraging/body mass columns.

parse_fish_traits *Parse FISHMORPH freshwater fish morphological traits (CSV)*

Description

Parse FISHMORPH freshwater fish morphological traits (CSV)

Usage

```
parse_fish_traits(path)
```

Arguments

path	Character. Path to FISHMORPH_Database.csv.

Value

data.frame with canonical_name + morphology columns.

parse_fishbase *Parse FishBase species and ecology tables (via rfishbase)*

Description

Pulls the species and ecology tables from rfishbase, joins them, and builds a canonical_name + trait data.frame.

Usage

```
parse_fishbase(path)
```

Arguments

path	Character. Not used (rfishbase fetches data directly), kept for interface consistency.

Value

data.frame with canonical_name + body/depth/diet columns.

parse_fungal_traits *Parse FungalTraits XLSX (Table S1, genus-level traits)*

Description

Reads the Polme et al. (2020) supplementary XLSX, selects the most informative trait columns, and returns a data.frame keyed on genus.

Usage

```
parse_fungal_traits(path)
```

Arguments

path	Character. Path to the downloaded XLSX file.

Value

data.frame with genus + 9 trait columns.

parse_funguild *Parse FUNGuild JSON database dump*

Description

Reads the JSON array returned by the FUNGuild API, filters to genus and species-level entries, and returns a clean data.frame with canonical_name and trait columns.

Usage

```
parse_funguild(path)
```

Arguments

path	Character. Path to the downloaded JSON/HTML file.

Value

data.frame with canonical_name + trophic_mode + guild + growth_morphology + confidence_ranking.

parse_gbif_common_names *Parse GBIF vernacular names (VernacularName.tsv + Taxon.tsv)*

Description

Parse GBIF vernacular names (VernacularName.tsv + Taxon.tsv)

Usage

```
parse_gbif_common_names(dir_path)
```

Arguments

dir_path	Character. Directory holding VernacularName.tsv and Taxon.tsv.

Value

data.frame with canonical_name + lang + common_name.

parse_glonaf *Parse GloNAF taxon-region data (multiple CSVs/XLSXs from ZIP)*

Description

Parse GloNAF taxon-region data (multiple CSVs/XLSXs from ZIP)

Usage

```
parse_glonaf(dir_path)
```

Arguments

dir_path	Character. Directory containing the GloNAF files.

Value

data.frame with canonical_name + region_id + naturalized.

parse_griis *Parse GRIIS Country Compendium CSV*

Description

Parse GRIIS Country Compendium CSV

Usage

```
parse_griis(path)
```

Arguments

path	Character. Path to GRIIS_Country_Compendium_V1_0.csv.

Value

data.frame with canonical_name + country_code + invasive_status.

parse_leda *Parse LEDA trait files (multiple semicolon/tab-delimited files)*

Description

Parse LEDA trait files (multiple semicolon/tab-delimited files)

Usage

```
parse_leda(dir_path)
```

Arguments

dir_path	Character. Directory containing the LEDA *.txt files.

Value

data.frame with canonical_name + 10 LEDA trait columns.

parse_leptraits *Parse LepTraits 1.0 butterfly consensus CSV*

Description

Parse LepTraits 1.0 butterfly consensus CSV

Usage

```
parse_leptraits(path)
```

Arguments

path	Character. Path to consensus.csv.

Value

data.frame with canonical_name + butterfly trait columns.

parse_lizard_traits *Parse Meiri (2018) lizard traits (XLSX from Figshare)*

Description

Parse Meiri (2018) lizard traits (XLSX from Figshare)

Usage

```
parse_lizard_traits(path)
```

Arguments

path	Character. Path to the ReptTraits/Meiri XLSX (or CSV/TSV).

Value

data.frame with canonical_name + lizard trait columns.

parse_ncbi_common_names *Parse NCBI common names from names.dmp*

Description

Parse NCBI common names from names.dmp

Usage

```
parse_ncbi_common_names(dir_path)
```

Arguments

dir_path	Character. Directory containing names.dmp.

Value

data.frame with canonical_name + lang (NA) + common_name.

parse_ott_common_names *Parse OTT common names from synonyms.tsv + taxonomy.tsv*

Description

Parse OTT common names from synonyms.tsv + taxonomy.tsv

Usage

```
parse_ott_common_names(dir_path)
```

Arguments

dir_path	Character. Directory containing synonyms.tsv + taxonomy.tsv.

Value

data.frame with canonical_name + lang (NA) + common_name.

parse_pantheria *Parse PanTHERIA mammal life-history traits (TSV)*

Description

Parse PanTHERIA mammal life-history traits (TSV)

Usage

```
parse_pantheria(path)
```

Arguments

path	Character. Path to PanTHERIA.txt.

Value

data.frame with canonical_name + life-history columns.

parse_wcvp *Parse WCVP names + distribution (from extracted ZIP directory)*

Description

Parse WCVP names + distribution (from extracted ZIP directory)

Usage

```
parse_wcvp(dir_path)
```

Arguments

dir_path	Character. Directory containing wcvp_names + wcvp_distribution.

Value

data.frame with canonical_name + tdwg_code + native_status.

parse_woodiness *Parse Zanne et al. 2014 woodiness CSV*

Description

Parse Zanne et al. 2014 woodiness CSV

Usage

```
parse_woodiness(path)
```

Arguments

path	Character. Path to GlobalWoodinessDatabase.csv.

Value

data.frame with canonical_name + woodiness.

precompute_backbone *Full precompute pipeline: keys + synonym embedding*

Description

Adds the matching-key columns and embedded synonym info that taxify's runtime expects. Operates on a normalized backbone (column names from `normalize_backbone()`).

Usage

```
precompute_backbone(df, synonym_pattern = "SYNONYM")
```

Arguments

df	A normalized backbone data.frame.
synonym_pattern	Regex for synonym detection.

Value

The data.frame ready for `build_vtr()`.

publish_release *Create a GitHub release and upload backbone artifacts*

Description

Uses the gh CLI. Assumes gh is authenticated and on PATH.

Usage

```
publish_release(
  backend_name,
  version,
  vtr_path,
  delta_path = NULL,
  meta_path = NULL,
  extras = character(0L),
  repo = "gcol33/taxifydb",
  notes = NULL
)
```

Arguments

backend_name	Character. Backend identifier.
version	Character. Version string.
vtr_path	Character. Path to the main .vtr file.
delta_path	Character or NULL. Path to the .xdelta file.
meta_path	Character or NULL. Path to the .meta sidecar.
extras	Character vector. Paths to additional sidecar artifacts (e.g., col_species_profile.vtr) that should be uploaded with the release and recorded in the manifest. Must exist on disk; basenames are used as the manifest entry names.
repo	Character. GitHub repo (e.g., "gcol33/taxifydb").
notes	Character. Release notes.

Value

The release tag (invisibly).

read_algaebase *Read and normalize a fetched list of AlgaeBase records*

Description

Read and normalize a fetched list of AlgaeBase records

Usage

```
read_algaebase(records, verbose = TRUE)
```

Arguments

records	A list of raw search records (from download_algaebase()).
verbose	Logical.

Value

A normalized data.frame.

read_col *Read and normalize the COL Taxon.tsv*

Description

Read and normalize the COL Taxon.tsv

Usage

```
read_col(col_dir, verbose = TRUE)
```

Arguments

col_dir	Character. Path to the extracted COL directory.
verbose	Logical.

Value

A normalized data.frame.

read_dmp *Read a pipe-delimited NCBI .dmp file*

Description

Read a pipe-delimited NCBI .dmp file

Usage

```
read_dmp(path, col_names)
```

Arguments

path	Character. Path to the .dmp file.
col_names	Character vector of column names.

Value

A data.frame.

read_euromed *Read and normalize the Euro+Med CSV*

Description

Resolves the parent-child hierarchy for family/genus, maps synonym relationships via TaxonConceptID, and produces the unified backbone schema.

Usage

```
read_euromed(csv_path, verbose = TRUE)
```

Arguments

csv_path	Character. Path to the EuroMed.csv file.
verbose	Logical.

Value

A normalized data.frame.

read_fungorum *Read and normalize the Species Fungorum Plus ColDP files*

Description

Read and normalize the Species Fungorum Plus ColDP files

Usage

```
read_fungorum(fungorum_dir, verbose = TRUE)
```

Arguments

fungorum_dir	Character. Directory containing Taxon.tsv, Name.tsv, Synonym.tsv.
verbose	Logical.

Value

A normalized data.frame.

read_gbif *Read and normalize the GBIF backbone*

Description

Read and normalize the GBIF backbone

Usage

```
read_gbif(gz_path, verbose = TRUE)
```

Arguments

gz_path	Character. Path to simple.txt.gz.
verbose	Logical.

Value

A normalized data.frame ready for precompute_backbone().

read_itis *Read and normalize the ITIS SQLite database*

Description

Read and normalize the ITIS SQLite database

Usage

```
read_itis(sqlite_path, verbose = TRUE)
```

Arguments

sqlite_path	Character. Path to the ITIS .sqlite file.
verbose	Logical.

Value

A normalized data.frame ready for precompute_backbone().

read_ncbi *Read and normalize the NCBI taxonomy dump*

Description

Read and normalize the NCBI taxonomy dump

Usage

```
read_ncbi(dump_dir, verbose = TRUE)
```

Arguments

dump_dir	Character. Path to the extracted taxdump directory.
verbose	Logical.

Value

A normalized data.frame ready for precompute_backbone().

read_ott *Read and normalize the OTT taxonomy*

Description

Read and normalize the OTT taxonomy

Usage

```
read_ott(ott_dir, verbose = TRUE)
```

Arguments

ott_dir	Character. Path to the extracted OTT directory.
verbose	Logical.

Value

A normalized data.frame.

read_wfo *Read and normalize the WFO classification file*

Description

Read and normalize the WFO classification file

Usage

```
read_wfo(txt_path, verbose = TRUE)
```

Arguments

txt_path	Character. Path to the WFO classification.txt file.
verbose	Logical.

Value

A normalized data.frame ready for `precompute_backbone()`.

read_worms *Read and normalize the WoRMS taxonomy*

Description

Read and normalize the WoRMS taxonomy

Usage

```
read_worms(worms_dir, verbose = TRUE)
```

Arguments

worms_dir	Character. Path to the extracted DwC-A directory.
verbose	Logical.

Value

A normalized data.frame.

resolve_enrichment_names *Resolve enrichment names against all taxify backends*

Description

Takes an enrichment data.frame with canonical_name + trait columns and expands it so that each source name maps to all unique accepted_name values across the requested backends.

Usage

```
resolve_enrichment_names(
  df,
  group_cols = NULL,
  backends = c("wfo", "col", "gbif", "itis", "ncbi", "ott", "worms"),
  verbose = TRUE,
  use_lookup = TRUE
)
```

Arguments

df	A data.frame with at least a canonical_name column.
group_cols	Character vector of grouping columns. Deduplication uses canonical_name + group_cols as the key. Default NULL.
backends	Character vector of backend names. Default: all 7.
verbose	Logical.
use_lookup	Logical. Try the hash-join fast path first. Default TRUE.

Details

By default tries the fast hash-join path against per-backend `name_lookup.vtr` files in the user's taxify data directory (built by `build_all_name_lookups()`); falls back to per-name-per-backend `taxify::taxify()` if no lookup files are found.

Value

The expanded data.frame.

`resolve_hierarchy` *Resolve family/genus by walking a parent-child hierarchy*

Description

Many backends (ITIS, NCBI, OTT) store taxonomy as a parent-child tree without explicit family/genus columns. This function walks up the tree to find the nearest ancestor at each target rank.

Usage

```
resolve_hierarchy(df, target_ranks = c("family", "genus"), max_depth = 20L)
```

Arguments

`df`	A data.frame with columns `id`, `parent_id`, `rank`, `name`.
`target_ranks`	Character vector of ranks to resolve (e.g., `c("family", "genus")`).
`max_depth`	Maximum number of hops (default 20).

Value

The input data.frame with new columns named after target_ranks, containing the resolved ancestor name at that rank.

`sha256` *Compute SHA-256 checksum of a file*

Description

Compute SHA-256 checksum of a file

Usage

```
sha256(path)
```

Arguments

`path`	Character. Path to the file.

Value

Character. Hex-encoded SHA-256 hash.

`update_enrichment_manifest` *Update manifest.json enrichment entry from built meta.json*

Description

Update manifest.json enrichment entry from built meta.json

Usage

```
update_enrichment_manifest(
  manifest_path,
  name,
  vtr_path,
  repo = "gcol33/taxifydb"
)
```

Arguments

`manifest_path`	Character. Path to manifest.json.
`name`	Character. Enrichment identifier.
`vtr_path`	Character. Path to the built .vtr file.
`repo`	Character. GitHub repo for URL construction.

Value

The updated manifest (invisibly).

`update_manifest` *Update manifest.json after a successful backbone build*

Description

Update manifest.json after a successful backbone build

Usage

```
update_manifest(
  manifest_path,
  backend_name,
  version,
  vtr_path,
  delta_path = NULL,
  delta_from = NULL,
  extras = character(0L),
  repo = "gcol33/taxifydb",
  source_url = NULL
)
```

Arguments

manifest_path	Character. Path to manifest.json.
backend_name	Character.
version	Character.
vtr_path	Character.
delta_path	Character or NULL.
delta_from	Character or NULL. Previous version the delta is from.
extras	Character vector. Paths to sidecar artifacts uploaded with the release. Recorded as `extras: [{name, url, size, sha256}]` in the manifest entry; the runtime downloader fetches each into the same versioned directory as the main `.vtr`.
repo	Character. GitHub repo for URL construction.
source_url	Character. Original data source URL.

Value

The updated manifest (invisibly).

Index