Package: traits.build (via r-universe)

December 20, 2024

Type Package

Title A workflow for harmonising trait data from diverse sources into a documented standard structure

Version 2.0.0

Maintainer Daniel Falster <daniel.falster@unsw.edu.au>

Description The `traits.build` package provides a workflow to harmonise trait data from diverse sources. The code was originally built to support AusTraits (see Falster et al 2021, <doi:10.1038/s41597-021-01006-6>, <https://github.com/traitecoevo/autraits.build>) and has been generalised here to support construction of other trait databases. For detailed instructions and examples see <https://traitecoevo.github.io/traits.build-book/>.

BugReports https://github.com/traitecoevo/traits.build/issues

URL http://traitecoevo.github.io/traits.build/

License BSD_2_clause + file LICENCE

Depends R (>= 4.3.0), base, dplyr, lubridate, readr, stringr, tidyr, austraits

Imports crayon, git2r, kableExtra, magrittr, purrr, forcats, RefManageR, rlang, rmarkdown, stringi, styler, testthat, tibble, whisker, yaml, lifecycle

Suggests furrr, remake, leaflet, bibtex, knitr, bench, devtools, markdown, pkgdown, rcrossref, ggplot2, ggbeeswarm, gridExtra, scales, zip, covr

Remotes richfitz/remake, traitecoevo/austraits

Encoding UTF-8

VignetteBuilder knitr

RoxygenNote 7.3.2

Config/testthat/edition 3

Roxygen list(markdown = TRUE)

Config/pak/sysreqs libfontconfig1-dev libfreetype6-dev make libgit2-dev libicu-dev libpng-dev libxml2-dev libssl-dev libx11-dev

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

RemoteUrl https://github.com/traitecoevo/traits.build

RemoteRef HEAD

RemoteSha e15d2a1ad72f0a5c6f1339a22d3fff036b9cd6a0

Contents

bib_print *Format BibEntry using RefManageR*

Description

Format BibEntry object according to desired style using RefManageR

Usage

```
bib_print(
  bib,
  .opts = list(first.inits = TRUE, max.names = 1000, style = "markdown")
)
```

Arguments

bib	BibEntry object
.opts	List of parameters for formatting style

Value

Character string of formatted reference

build_add_version *Add version information to AusTraits*

Description

Add version information to AusTraits

Usage

```
build_add_version(austraits, version, git_sha)
```

Arguments

austraits	AusTraits database object
version	Version number
git_sha	Git SHA

Value

AusTraits database object with version information added

build_combine *Combine all the AusTraits studies into the compiled AusTraits database*

Description

`build_combine` compiles all the loaded studies into a single AusTraits database object as a large list.

[Deprecated]

Usage

```
build_combine(..., d = list(...))
```

Arguments

`...`	Arguments passed to other functions
`d`	List of all the AusTraits studies

Value

AusTraits compilation database as a large list

build_setup_pipeline *Update the* `remake.yml` *file with new studies*

Description

`build_setup_pipeline` rewrites the `remake.yml` file to include new studies.

Usage

```
build_setup_pipeline(
  dataset_ids = dir("data"),
  method = "base",
  database_name = "database",
  template = select_pipeline_template(method),
  workers = 1
)
```

Arguments

`dataset_ids`	`dataset_id`'s to include; by default includes all
`method`	Approach to use in build
`database_name`	Name of database to be built
`template`	Template used to build
`workers`	Number of workers/parallel processes to use when using method = "furrr"

Value

Updated `remake.yml` file

check_pivot_duplicates *Identify duplicates preventing pivoting wider*

Description

Identify duplicates preventing pivoting wider

Usage

```
check_pivot_duplicates(
  database_object,
  dataset_ids = unique(database_object$traits$dataset_id)
)
```

Arguments

`database_object`	Database object
`dataset_ids`	`dataset_id`'s to check for duplicates; default is all of them

Value

Tibble with duplicates and pivot columns

check_pivot_wider *Test whether a dataset can pivot wider*

Description

Test whether the traits table of a dataset can pivot wider with the minimum required columns.

Usage

```
check_pivot_wider(dataset)
```

Arguments

`dataset`	Built dataset with `test_build_dataset`

Value

Number of rows with duplicates preventing pivoting wider

create_tree_branch *Format a tree structure from a vector*

Description

create_tree_branch() is used to create a tree structure to show how things are related. In AusTraits, this is used in the vignettes to show the file structure of the repository and also to show the different components of the AusTraits database.

Usage

```
create_tree_branch(x, title, prefix = "")
```

Arguments

x	Vector of terms
title	Name of branch
prefix	Specifies the amount of indentation

Value

Vector of character strings for the tree structure

dataset_build *Build dataset*

Description

Build specified dataset. This function completes three steps, which can be executed separately if desired: dataset_configure, dataset_process, dataset_update_taxonomy

Usage

```
dataset_build(
  filename_metadata,
  filename_data_raw,
  definitions,
  unit_conversion_functions,
  schema,
  resource_metadata,
  taxon_list,
  filter_missing_values = TRUE
)
```

Arguments

filename_metadata
: Metadata yaml file for a given study

filename_data_raw
: Raw data.csv file for any given study

definitions
: Definitions read in from the traits.yml

unit_conversion_functions
: unit_conversion.csv file read in from the config folder

schema
: Schema for traits.build

resource_metadata
: metadata for the compilation

taxon_list
: Taxon list

filter_missing_values
: Default filters missing values from the excluded data table; change to false to see the rows with missing values

Value

List, AusTraits database object

Examples

```
## Not run:
dataset_build(
  "data/Falster_2003/data.csv",
  "data/Falster_2003/metadata.yml",
  read_yaml("config/traits.yml"),
   get_unit_conversions("config/unit_conversions.csv"),
   get_schema(),
   get_schema("config/metadata.yml", "metadata"),
   read_csv_char("config/taxon_list.csv")
)

## End(Not run)
```

dataset_configure *Configure AusTraits database object*

Description

Creates the config object which gets passed onto dataset_process. The config list contains the subset of definitions and unit conversions for those traits for a each study. dataset_configure is used in the remake::make process to configure individual studies mapping the individual traits found in that study along with any relevant unit conversions and definitions. dataset_configure and dataset_process are applied to every study in the remake.yml file.

Usage

```
dataset_configure(filename_metadata, definitions)
```

Arguments

`filename_metadata`
Metadata yaml file for a given study

`definitions` Definitions read in from the `traits.yml`

Value

List with `dataset_id`, `metadata`, `definitions` and `unit_conversion_functions`

Examples

```
## Not run:
dataset_configure("data/Falster_2003/metadata.yml", read_yaml("config/traits.yml"))

## End(Not run)
```

`dataset_find_taxon` *Find list of unique datasets within compilation containing specified taxa*

Description

Find list of unique datasets within compilation containing specified taxa

Usage

```
dataset_find_taxon(taxa, austraits, original_name = FALSE)
```

Arguments

`taxa` A vector which contains species names

`austraits` AusTraits compilation

`original_name` Logical; if TRUE use column in compilation which contains original species names, default = FALSE

Value

List of unique datasets within compilation containing each taxon

dataset_process *Load Dataset*

Description

dataset_process is used to load individual studies using the config file generated from dataset_configure(). dataset_configure and dataset_process are applied to every study in the remake.yml file.

Usage

```
dataset_process(
  filename_data_raw,
  config_for_dataset,
  schema,
  resource_metadata,
  unit_conversion_functions,
  filter_missing_values = TRUE
)
```

Arguments

filename_data_raw
Raw data.csv file for any given study

config_for_dataset
Config settings generated from dataset_configure()

schema
Schema for traits.build

resource_metadata
Metadata about the traits compilation read in from the config folder

unit_conversion_functions
unit_conversion.csv file read in from the config folder

filter_missing_values
Default filters missing values from the excluded data table; change to false to see the rows with missing values

Value

List, AusTraits database object

Examples

```
## Not run:
dataset_process("data/Falster_2003/data.csv", dataset_configure("data/Falster_2003/metadata.yml",
read_yaml("config/traits.yml")),
get_schema(),
get_schema("config/metadata.yml", "metadata"),
get_unit_conversions("config/unit_conversions.csv"))

## End(Not run)
```

dataset_report *Build reports for listed datasets*

Description

Builds a detailed report for every dataset with a unique `dataset_id`, based on the template Rmd file provided. The reports are rendered as html files and saved in the specified output folder.

Usage

```
dataset_report(
  dataset_id,
  austraits,
  overwrite = FALSE,
  output_path = "export/reports",
 input_file = system.file("support", "report_dataset.Rmd", package = "traits.build"),
  quiet = TRUE,
  keep = FALSE
)
```

Arguments

`dataset_id`	Name of specific study/dataset
`austraits`	Compiled austraits database
`overwrite`	Logical value to determine whether to overwrite existing report
`output_path`	Location where rendered report will be saved
`input_file`	Report script (.Rmd) file to build study report
`quiet`	An option to suppress printing during rendering from knitr, pandoc command line and others
`keep`	Keep intermediate Rmd file used?

Value

Html file of the rendered report located in the specified output folder

dataset_test *Test whether specified* `dataset_id` *has the correct setup*

Description

Run tests to ensure that specified `dataset_id` has the correct setup.

Usage

```
dataset_test(
  dataset_ids,
  path_config = "config",
  path_data = "data",
  reporter = testthat::CompactProgressReporter
)
```

Arguments

dataset_ids	Vector of dataset_id for sources to be tested
path_config	Path to folder containing configuration files
path_data	Path to folder containing data files
reporter	testthat reporter to use to summarise output

dataset_test_worker *Test whether specified* dataset_id *has the correct setup*

Description

Run tests to ensure that specified dataset_id has the correct setup.

Usage

```
dataset_test_worker(
  test_dataset_ids,
  path_config = "config",
  path_data = "data",
  schema = get_schema(),
  definitions = get_schema(file.path(path_config, "traits.yml"), I("traits"))
)
```

Arguments

test_dataset_ids	Vector of dataset_id for sources to be tested
path_config	Path to folder containing configuration files
path_data	Path to folder containing data files
schema	Data schema
definitions	Trait defininitons

`dataset_update_taxonomy` *Apply taxonomic updates to austraits_raw*

Description

Applies taxonomic updates to austraits_raw.

Usage

```
dataset_update_taxonomy(austraits_raw, taxa)
```

Arguments

`austraits_raw`	AusTraits compiled data as a large list without taxonomic updates applied
`taxa`	Taxon list

Value

List of AusTraits compiled data with taxonomic updates applied

`get_schema` *Load schema for an traits.build data compilation (excluding traits)*

Description

Load schema for an traits.build data compilation (excluding traits)

Usage

```
get_schema(
  path = system.file("support", "traits.build_schema.yml", package = "traits.build"),
  subsection = NULL
)
```

Arguments

`path`	path to schema file. By default loads version included with the package
`subsection`	section to load

Value

a list

Examples

```
{

schema <- get_schema()
}
```

---

get_unit_conversions *Make unit conversion functions*

Description

Make unit conversion functions

Usage

```
get_unit_conversions(filename)
```

Arguments

filename	Name of file containing unit conversions

Value

List of conversion functions

Examples

```
## Not run:
get_unit_conversions("config/unit_conversions.csv")

## End(Not run)
```

---

metadata_add_contexts *For specified* dataset_id *import context data from a dataframe*

Description

This functions asks users which columns in the dataframe they would like to keep and records this appropriately in the metadata. The input data is assumed to be in wide format. The output may require additional manual editing.

Usage

```
metadata_add_contexts(dataset_id, overwrite = FALSE, user_responses = NULL)
```

Arguments

dataset_id Identifier for a particular study in the database

overwrite Overwrite existing information

user_responses Named list containing simulated user input for manual selection of variables, mainly for testing purposes

---

metadata_add_locations

For specified `dataset_id` *import location data from a dataframe*

Description

This functions asks users which columns in the dataframe they would like to keep and records this appropriately in the metadata. The input data is assumed to be in wide format. The output may require additional manual editing.

Usage

```
metadata_add_locations(dataset_id, location_data, user_responses = NULL)
```

Arguments

dataset_id Identifier for a particular study in the database

location_data A dataframe of site variables

user_responses Named list containing simulated user input for manual selection of variables, mainly for testing purposes

Examples

```
## Not run:
austraits$locations %>% dplyr::filter(dataset_id == "Falster_2005_1") %>%
select(-dataset_id) %>% spread(location_property, value) %>% type_convert() -> location_data
metadata_add_locations("Falster_2005_1", location_data)

## End(Not run)
```

metadata_add_source_bibtex

Adds citation details to a metadata file for given study

Description

Adds citation details to a metadata file for given study

Usage

```
metadata_add_source_bibtex(
  dataset_id,
  file,
  type = "primary",
  drop = c("dateobj", "month")
)
```

Arguments

`dataset_id`	Identifier for a particular study in the database
`file`	Name of file where reference is saved
`type`	Type of reference: `primary`, `secondary` or `original` (or `original_01`, `original_02`, etc., for multiple sources)
`drop`	Variables in bibtex to ignore

Value

`metadata.yml` file with citation details added

metadata_add_source_doi

Adds citation details from a doi to a metadata file for a `dataset_id`

Description

Uses rcrossref package to access publication details from the crossref database

Usage

```
metadata_add_source_doi(..., doi, bib = NULL)
```

Arguments

`...`	Arguments passed from metadata_add_source_bibtex()
`doi`	doi of reference to add
`bib`	(Only use for testing purposes) Result of calling `bib rcrossref::cr_cn(doi)`

Value

`metadata.yml` file with citation details added

---

metadata_add_substitution

Add a categorical trait value substitution into a metadata file for a `dataset_id`

Description

`metadata_add_substitution` is used to align the categorical trait values used by a contributor to the categorical values supported by the database. These values are defined in the `traits.yml` file.

Usage

```
metadata_add_substitution(dataset_id, trait_name, find, replace)
```

Arguments

`dataset_id`	Identifier for a particular study in the database
`trait_name`	The database defined name for a particular trait
`find`	Trait value in the original data.csv file
`replace`	Trait value supported by database

Value

`metadata.yml` file with a substitution added

---

metadata_add_substitutions_list

Add a dataframe of trait value substitutions into a metadata file for a dataset_id

---

Description

Add a dataframe of trait value substitutions into a metadata file for a dataset_id

Usage

```
metadata_add_substitutions_list(dataset_id, substitutions)
```

Arguments

`dataset_id`	Identifier for a particular study in the database
`substitutions`	Dataframe of trait value substitutions

Value

metadata.yml file with multiple trait value substitutions added

---

metadata_add_substitutions_table

Substitutions from a dataframe

Description

Function that simultaneously adds many trait value replacements, potentially across many trait_name's and dataset_id's, to the respective metadata.yml files. This function will be used to quickly re-align/re-assign trait values across all studies.

Usage

```
metadata_add_substitutions_table(
  dataframe_of_substitutions,
  dataset_id,
  trait_name,
  find,
  replace
)
```

Arguments

dataframe_of_substitutions
: Dataframe with columns indicating dataset_id, trait_name, original trait values (find), and database aligned trait value (replace)

dataset_id
: Name of column containing study dataset_id(s) in database

trait_name
: Name of column containing trait name(s) for which a trait value replacement needs to be made

find
: Name of column containing trait values submitted by the contributor for a data observation

replace
: Name of column containing database aligned trait values

Value

Modified metadata files with trait value replacements

Examples

```
## Not run:
read_csv("export/dispersal_syndrome_substitutions.csv") %>%
  select(-extra) %>%
  filter(dataset_id == "Angevin_2011") -> dataframe_of_substitutions
metadata_add_substitutions_table(dataframe_of_substitutions, dataset_id, trait_name, find, replace)

## End(Not run)
```

metadata_add_taxonomic_change

Add a taxonomic change into the `metadata.yml` *file for a* `dataset_id`

Description

Add a single taxonomic change into the `metadata.yml` file for a specific study.

Usage

```
metadata_add_taxonomic_change(
  dataset_id,
  find,
  replace,
  reason,
  taxonomic_resolution,
  overwrite = TRUE
)
```

Arguments

`dataset_id`	Identifier for a particular study in the database
`find`	Original name used by the contributor
`replace`	Taxonomic name accepted by APC or APNI
`reason`	Reason for taxonomic change
`taxonomic_resolution`	The rank of the most specific taxon name (or scientific name) to which a submitted orignal name resolves
`overwrite`	Parameter indicating whether preexisting find-replace entries should be overwritten. Defaults to `true`

Value

`metadata.yml` file with taxonomic change added

metadata_add_taxonomic_changes_list

Add a list of taxonomic updates into a metadata file for a `dataset_id`

Description

Add multiple taxonomic changes to the `metadata.yml` file using a dataframe containing the taxonomic changes to be made.

Usage

```
metadata_add_taxonomic_changes_list(dataset_id, taxonomic_updates)
```

Arguments

`dataset_id` Identifier for a particular study in the database

`taxonomic_updates`
Dataframe of taxonomic updates

Value

`metadata.yml` file with multiple taxonomic updates added

`metadata_add_traits` *For specified* `dataset_id`*, populate columns for traits into metadata*

Description

This function asks users which traits they would like to keep, and adds a template for those traits in the metadata. This template must then be finished manually.

Usage

```
metadata_add_traits(dataset_id, user_responses = NULL)
```

Arguments

`dataset_id` Identifier for a particular study in the database

`user_responses` Named list containing simulated user input for manual selection of variables, mainly for testing purposes

Details

Can also be used to add a trait to an existing metadata file.

metadata_check_custom_R_code

Check the output of running custom_R_code *specified in the metadata for specified* dataset_id

Description

Function to check the output of running custom_R_code specified in the metadata.yml file for specified dataset_id. For the specified dataset_id, reads in the file data.csv and applies manipulations as described in the file metadata.yml

Usage

```
metadata_check_custom_R_code(dataset_id, path_data = "data")
```

Arguments

dataset_id	Identifier for a particular study in the database
path_data	Path to folder with data

metadata_create_template

Create a template of file metadata.yml *for specified* dataset_id

Description

Includes place-holders for major sections of the metadata.

Usage

```
metadata_create_template(
  dataset_id,
  path = file.path("data", dataset_id),
  skip_manual = FALSE,
  user_responses = NULL
)
```

Arguments

dataset_id	Identifier for a particular study in the database
path	Location of file where output is saved
skip_manual	Allows skipping of manual selection of variables, default = FALSE
user_responses	Named list containing simulated user input for manual selection of variables, mainly for testing purposes

Value

A yaml file template for metadata

`metadata_exclude_observations` *Exclude observations in a yaml file for a* `dataset_id`

Description

Exclude observations in a yaml file for a `dataset_id`

Usage

```
metadata_exclude_observations(dataset_id, variable, find, reason)
```

Arguments

`dataset_id`	Identifier for a particular study in the database
`variable`	Variable name
`find`	Term to find by
`reason`	Reason for exclusion

Value

`metadata.yml` file with excluded observations

`metadata_find_taxonomic_change` *Find* `dataset_id`*'s with a given taxonomic change*

Description

Find `dataset_id`'s with a given taxonomic change

Usage

```
metadata_find_taxonomic_change(find, replace = NULL, studies = NULL)
```

Arguments

`find`	Name of original species
`replace`	Name of replacement species, default = NULL
`studies`	Name of studies to look through, default = NULL

`metadata_path_dataset_id` *Path to the* `metadata.yml` *file for specified* `dataset_id`

Description

Path to the `metadata.yml` file for specified `dataset_id`

Usage

```
metadata_path_dataset_id(dataset_id, path_data = "data")
```

Arguments

`dataset_id`	Identifier for a particular study in the database
`path_data`	Path to folder with data

Value

A string

`metadata_remove_taxonomic_change` *Remove a taxonomic change from a yaml file for a* `dataset_id`

Description

Remove a taxonomic change from a yaml file for a `dataset_id`

Usage

```
metadata_remove_taxonomic_change(dataset_id, find)
```

Arguments

`dataset_id`	Identifier for a particular study in the database
`find`	Taxonomic name to find

Value

`metadata.yml` file with a taxonomic change removed

metadata_update_taxonomic_change *Update a taxonomic change into a yaml file for a* dataset_id

Description

Update a taxonomic change into a yaml file for a `dataset_id`

Usage

```
metadata_update_taxonomic_change(
  dataset_id,
  find,
  replace,
  reason,
  taxonomic_resolution
)
```

Arguments

`dataset_id`	Identifier for a particular study in the database
`find`	Original taxonomic name
`replace`	Updated taxonomic name to replace original taxonomic name
`reason`	Reason for change
`taxonomic_resolution`	The rank of the most specific taxon name (or scientific name) to which a submitted orignal name resolves

Value

`metadata.yml` file with added substitution

metadata_user_select_column *Select column by user*

Description

`metadata_user_select_column` is used to select which columns in a dataframe/tibble corresponds to the variable of interest. It is used to compile the metadata yaml file by prompting the user to choose the relevant columns. It is used in `metadata_add_locations` and `metadata_create_template`.

Usage

```
metadata_user_select_column(column, choices)
```

Arguments

`column`	Name of the variable of interest
`choices`	The options that can be selected from

`metadata_user_select_names` *Select variable names by user*

Description

`metadata_user_select_names` is used to prompt the user to select the variables that are relevant for compiling the metadata yaml file. It is currently used for `metadata_add_traits`, `metadata_add_locations` and `metadata_add_contexts`.

Usage

```
metadata_user_select_names(title, vars)
```

Arguments

`title`	Character string providing the instruction for the user
`vars`	Variable names

`notes_random_string` *Create a string of random letters*

Description

Creates a string of random letters with 8 characters as the default, useful for defining unique hyperlinks

Usage

```
notes_random_string(n = 8)
```

Arguments

`n`	numerical integer, default is 8

Value

character string with 8 letters

notetaker_add_note *Add a note to the note recorder as a new row*

Description

Add a note to the note recorder as a new row

Usage

```
notetaker_add_note(notes, new_note)
```

Arguments

`notes`	object containing the report notes
`new_note`	vector of character notes to be added to existing notes

Value

A tibble with additional notes added

notetaker_as_note *Create a tibble with two columns with note and link*

Description

Creates a tibble with two columns with one column consisting of a randomly generated string of letters

Usage

```
notetaker_as_note(note, link = NA_character_)
```

Arguments

`note`	character string
`link`	character string, default is NA_character_ which generates a random string

Value

a tibble with two columns named note and link

`notetaker_get_note` *Return a specific row from notes*

Description

Returns a specific row from notes specified by i. Default is nrow(notes) which returns the last note

Usage

```
notetaker_get_note(notes, i = nrow(notes))
```

Arguments

`notes`	object containing the report notes
`i`	numerical; row number for corresponding note, default is nrow(notes)

Value

a single row from a tibble

`notetaker_print_all` *Print all notes*

Description

Print all notes

Usage

```
notetaker_print_all(notes, ..., numbered = TRUE)
```

Arguments

`notes`	object containing the report notes
`...`	arguments passed to other functions
`numbered`	logical default is TRUE

Value

character string containing the notes

notetaker_print_note *Print note (needs review?)*

Description

Print note (needs review?)

Usage

```
notetaker_print_note(
  note,
  as_anchor = FALSE,
  anchor_text = "",
  link_text = "link"
)
```

Arguments

`note`	object containing the report notes
`as_anchor`	logical default is FALSE
`anchor_text`	character string, default is ""
`link_text`	character string, default is "link"

Value

character string containing the notes

notetaker_print_notes *Print a specific row from notes*

Description

Prints a specific row from notes specified by i

Usage

```
notetaker_print_notes(notes, i = nrow(notes), ...)
```

Arguments

`notes`	object containing the report notes
`i`	specify the row which contains the note to be returned
`...`	arguments passed to notetaker_print_note()

Value

character string containing the notes

`notetaker_start` *Start note recorder (needs review?)*

Description

Note recorder used in report_study.Rmd file to initiate note recorder

Usage

```
notetaker_start()
```

Value

A tibble where notes are recorded

`process_add_all_columns` *Add or remove columns of data*

Description

Add or remove columns of data as needed so that all datasets have the same columns. Also adds in an error column.

Usage

```
process_add_all_columns(data, vars, add_error_column = TRUE)
```

Arguments

`data`	Dataframe containing study data read in as a csv file
`vars`	Vector of variable columns names to be included in the final formatted tibble
`add_error_column`	Adds an extra column called error if TRUE

Value

Tibble with the correct selection of columns including an error column

`process_convert_units` *Convert units to desired type*

Description

Convert units to desired type

Usage

```
process_convert_units(data, definitions, unit_conversion_functions)
```

Arguments

`data`	Tibble or dataframe containing the study data
`definitions`	Definitions read in from the `traits.yml` file in the config folder
`unit_conversion_functions`	`unit_conversions.csv` file stored in the config folder

Value

Tibble with converted units

`process_create_observation_id` *Create entity id*

Description

Creates 3-part entity id codes that combine a segment for species, population, and, when applicable, individual. This depends upon a `parsing_id` being established when the `data.csv` file is first read in.

Usage

```
process_create_observation_id(data, metadata)
```

Arguments

`data`	The traits table at the point where this function is called
`metadata`	Yaml file with metadata

Value

Character string

`process_custom_code` *Apply custom data manipulations*

Description

Applies custom data manipulations if the metadata field `custom_R_code` is not empty Otherwise no manipulations will be done by applying the `identity` function. The code `custom_R_code` assumes a single input.

Usage

```
process_custom_code(txt)
```

Arguments

`txt`	Character text within custom_R_code of a `metadata.yml` file

Value

character text containing custom_R_code if custom_R_code is not empty, otherwise no changes are made

`process_flag_excluded_observations` *Flag any excluded observations*

Description

Checks the metadata yaml file for any excluded observations. If there are none, returns the original data. If there are excluded observations returns the mutated data with excluded observations flagged in a new column.

Usage

```
process_flag_excluded_observations(data, metadata)
```

Arguments

`data`	Tibble or dataframe containing the study data
`metadata`	Yaml file with metadata

Value

Dataframe with flagged excluded observations if there are any

process_flag_out_of_range_values

Flag values outside of allowable range

Description

Flags any numeric values that are outside the allowable range defined in the `traits.yml` file.

Usage

```
process_flag_out_of_range_values(data, definitions)
```

Arguments

`data`	Tibble or dataframe containing the study data
`definitions`	Definitions read in from the `traits.yml` file in the config folder

Value

Tibble with flagged values outside of allowable range

process_flag_unsupported_characters

Flag values with unsupported characters

Description

Disallowed characters are flagged as errors, including for numeric traits, prior to unit conversions to avoid their conversion to NAs during the unit conversion process.

Usage

```
process_flag_unsupported_characters(data)
```

Arguments

`data`	Tibble or dataframe containing the study data

Value

Tibble with flagged values containing unsupported characters

`process_flag_unsupported_traits` *Flag any unrecognised traits*

Description

Flag any unrecognised traits, as defined in the `traits.yml` file.

Usage

```
process_flag_unsupported_traits(data, definitions)
```

Arguments

`data`	Tibble or dataframe containing the study data
`definitions`	Definitions read in from the `traits.yml` file in the config folder

Value

Tibble with unrecognised traits flagged in the "error" column

`process_flag_unsupported_values` *Flag disallowed trait values and disallowed characters*

Description

Flags any categorical traits values that are not on the list of allowed values defined in the `traits.yml` file. NA values are flagged as errors. Numeric values that cannot convert to numeric are also flagged as errors.

Usage

```
process_flag_unsupported_values(data, definitions)
```

Arguments

`data`	Tibble or dataframe containing the study data
`definitions`	Definitions read in from the `traits.yml` file in the config folder

Value

Tibble with flagged values that are unsupported categorical trait values, missing values or numeric trait values that cannot be converted to numeric

process_format_contexts

Format context data from list to tibble

Description

Format context data read in from the `metadata.yml` file. Converts from list to tibble.

Usage

```
process_format_contexts(my_list, dataset_id, traits)
```

Arguments

`my_list`	List of input information
`dataset_id`	Identifier for a particular study in the AusTraits database
`traits`	Table of trait data (for this function, just the data.csv file with custom_R_code applied)

Value

Tibble with context details if available

Examples

```
## Not run:
process_format_contexts(read_metadata("data/Apgaua_2017/metadata.yml")$context, dataset_id, traits)

## End(Not run)
```

process_format_contributors

Format contributors from list into tibble

Description

Format contributors, read in from the `metadata.yml` file. Converts from list to tibble.

Usage

```
process_format_contributors(my_list, dataset_id, schema)
```

Arguments

`my_list`	List of input information
`dataset_id`	Identifier for a particular study in the AusTraits database
`schema`	Schema for traits.build

Value

Tibble with details of contributors

Examples

```
## Not run:
process_format_contributors(read_metadata("data/Falster_2003/metadata.yml")$contributors)

## End(Not run)
```

`process_format_locations`

Format location data from list to tibble

Description

Format location data read in from the `metadata.yml` file. Converts from list to tibble.

Usage

```
process_format_locations(my_list, dataset_id, schema)
```

Arguments

`my_list`	List of input information
`dataset_id`	Identifier for a particular study in the AusTraits database
`schema`	Schema for traits.build

Value

Tibble with location details if available

Examples

```
## Not run:
process_format_locations(read_metadata("data/Falster_2003/metadata.yml")$locations, "Falster_2003")

## End(Not run)
```

process_generate_id *Function to generate sequence of integer ids from vector of names Determines number of 00s needed based on number of records*

Description

Function to generate sequence of integer ids from vector of names Determines number of 00s needed based on number of records

Usage

```
process_generate_id(x, prefix, sort = FALSE)
```

Arguments

`x`	Vector of text to convert
`prefix`	Text to put before id integer
`sort`	Logical to indicate whether x should be sorted before ids are generated

Value

Vector of ids

process_generate_method_ids *Function to generate sequence of integer ids for methods*

Description

Function to generate sequence of integer ids for methods

Usage

```
process_generate_method_ids(metadata_traits)
```

Arguments

`metadata_traits`	the traits section of the metadata

Value

Tibble with traits, methods, and method_id

process_parse_data *Process a single dataset*

Description

Process a single dataset with `dataset_id` using the associated `data.csv` and `metadata.yml` files. Adds a unique observation id for each row of observation, trait names are formatted using AusTraits accepted names and trait substitutions are added. `parse data` is used in the core workflow pipeline (i.e. in `load study`).

Usage

```
process_parse_data(data, dataset_id, metadata, contexts, schema)
```

Arguments

`data`	Tibble or dataframe containing the study data
`dataset_id`	Identifier for a particular study in the AusTraits database
`metadata`	Yaml file with metadata
`contexts`	Dataframe of contexts for this study
`schema`	Schema for traits.build

Value

Tibble in long format with AusTraits formatted trait names, trait substitutions and unique observation id added

process_standardise_names *Standardise species names*

Description

Enforces some standards on species names.

Usage

```
process_standardise_names(x)
```

Arguments

`x`	Vector, dataframe or list containing original species names

Value

Vector with standardised species names

process_taxonomic_updates

Apply taxonomic updates

Description

Applies taxonomic updates to the study data from the `metadata.yml` file.

Usage

```
process_taxonomic_updates(data, metadata)
```

Arguments

`data`	Tibble or dataframe containing the study data
`metadata`	Yaml file with metadata

Value

Tibble with the taxonomic updates applied

process_unit_conversion_name

Generate unit conversion name

Description

Creates the unit conversion name based on the original units and the units to be converted to.

Usage

```
process_unit_conversion_name(from, to)
```

Arguments

`from`	Character of original units
`to`	Character of units to be converted to

Value

Character string containing the name what units are being converted to

`read_csv_char` *Read in a csv as a tibble with column types as characters*

Description

Reads in a csv file using the `read_csv` function from readr with columns as characters.

Usage

```
read_csv_char(...)
```

Arguments

`...`	Arguments passed to the `read_csv()`

Value

A tibble

`read_metadata` *Read in a* `metadata.yml` *file for a study*

Description

Read in a `metadata.yml` file for a study

Usage

```
read_metadata(path)
```

Arguments

`path`	Location of the metadata file

read_metadata_dataset *Read the* metadata.yml *file for specified* dataset_id

Description

Read the metadata.yml file for specified dataset_id

Usage

```
read_metadata_dataset(dataset_id, path_data = "data")
```

Arguments

dataset_id	Identifier for a particular study in the database
path_data	Path to folder with data

Value

A list with contents of metadata for specified dataset_id

read_yaml *Read yaml (from package yaml)*

Description

Read yaml (from package yaml)

util_append_to_list *Add an item to the end of a list*

Description

Add an item to the end of a list

Usage

```
util_append_to_list(my_list, to_append)
```

Arguments

my_list	A list
to_append	A list

Value

A list merged with an added item at the end

Examples

```
 ## Not run:
util_append_to_list(as.list(dplyr::starwars)[c(1,2)], as.list(dplyr::starwars)[c(3,4)])

## End(Not run)
```

`util_bib_to_list` *Convert BibEntry object to a list*

Description

Convert BibEntry object to a list

Usage

```
util_bib_to_list(bib)
```

Arguments

`bib`	BibEntry object

Value

List

`util_check_all_values_in` *Check values in one vector against values in another vector*

Description

`util_check_all_values_in` checks if values in vector x are in y. Values in x may contain multiple values separated by `sep` so these are split first using `str_split`.

Usage

```
util_check_all_values_in(x, y, sep = " ")
```

Arguments

`x`	Vector
`y`	Vector
`sep`	Amount of space separating values to be split, default = " " (a single space)

Value

Vector of logical values

---

util_check_disallowed_chars

Check values in a vector do not contain disallowed characters

---

Description

`util_check_disallowed_chars` checks if values in a vector do not contain disallowed characters, i.e. values outside of ASCII.

Usage

```
util_check_disallowed_chars(object)
```

Arguments

`object`	Vector

Value

Vector of logical values

---

util_df_convert_character

Convert all columns in data frame to character

Description

Convert all columns in data frame to character

Usage

```
util_df_convert_character(df)
```

Arguments

`df`	A dataframe

Value

A dataframe

Examples

```
lapply(traits.build:::util_df_convert_character(dplyr::starwars), class)
```

`util_df_to_list` *Convert dataframe to list*

Description

Convert a dataframe to a named list, useful when converting to yaml.

[Deprecated]

Usage

```
util_df_to_list(df)
```

Arguments

`df`	A dataframe

Value

A (yaml) list

Examples

```
util_df_to_list(dplyr::starwars)
```

`util_extract_list_element` *Extract a trait element from the definitions$traits$elements*

Description

Extract a trait element from the definitions$traits$elements

Usage

```
util_extract_list_element(i, my_list, var)
```

Arguments

`i`	A value within the definitions$traits$elements list which refers to types of traits
`my_list`	The list that contains the element we're interested in (i.e. definitions$traits$elements)
`var`	The type of variable of a trait

Value

The element/properties of a trait

Examples

```
## Not run:
util_extract_list_element(1, definitions$traits$elements, "units")

## End(Not run)
```

---

util_get_SHA *Get SHA string from Github repository for latest commit*

Description

Get SHA string for the latest commit on Github for the repository. SHA is the abbreviated SHA-1 40 digit hexadecimal number which Github uses as the Commit ID to track changes made to a repo

Usage

```
util_get_SHA(path = ".")
```

Arguments

path	root directory where a specified file is located, default file name is the remake.yml file

Value

40-digit SHA character string for the latest commit to the repository

---

util_get_version *Retrieve version for compilation from definitions*

Description

Retrieve version for compilation from definitions

Usage

```
util_get_version(path = "config/metadata.yml")
```

Arguments

path	path to traits definitions

Value

a string

`util_kable_styling_html` *Format table with kable and default styling for html*

Description

Format table with kable and default styling for html

Usage

```
util_kable_styling_html(...)
```

Arguments

`...`	Arguments passed to `kableExtra::kable()`

`util_list_to_bib` *Convert a list of elements into a BibEntry object*

Description

Convert a list of elements into a BibEntry object

Usage

```
util_list_to_bib(ref)
```

Arguments

`ref`	List of elements for a reference

Value

BibEntry object

`util_list_to_df1` *Convert a list with single entries to dataframe*

Description

[Deprecated]

Usage

```
util_list_to_df1(my_list)
```

Arguments

`my_list`	A list with single entries

Value

A tibble with two columns

Examples

```
## Not run:
util_list_to_df1(as.list(dplyr::starwars)[2])

## End(Not run)
```

`util_list_to_df2` *Convert a list of lists to dataframe*

Description

Convert a list of lists to dataframe; requires that every list have same named elements.

[Deprecated]

Usage

```
util_list_to_df2(my_list, as_character = TRUE, on_empty = NA)
```

Arguments

`my_list`	A list of lists to dataframe
`as_character`	A logical value, indicating whether the values are read as character
`on_empty`	Value to return if my_list is NULL, NA or is length == 0, default = NA

Examples

```
util_list_to_df2(util_df_to_list(dplyr::starwars))
```

`util_replace_null` *Convert NULL values to a different value*

Description

`util_replace_null` converts NULL values a different value. Default is converting NULL to NA.

Usage

```
util_replace_null(x, val = NA)
```

Arguments

`x`	A NULL value or a non-NULL object
`val`	Specify what the null value should be returned as, default is NA

Value

NA or a non-NULL object

Examples

```
## Not run:
util_replace_null(NULL)

## End(Not run)
```

`util_separate_and_sort` *Split and sort cells with multiple values*

Description

`util_separate_and_sort`: For a vector x in which individual cell may have multiple values (separated by 'sep'), sort records within each cell alphabetically.

Usage

```
util_separate_and_sort(x, sep = " ")
```

Arguments

`x`	An individual cell with multiple values
`sep`	A separator, a whitespace is the default

Value

A vector of alphabetically sorted records

Examples

```
## Not run: util_separate_and_sort("z y x")
```

`util_standardise_doi` *Standardise doi*

Description

Standardise doi

Usage

```
util_standardise_doi(doi)
```

Arguments

`doi`	doi of reference to add

`write_metadata` *Write* `metadata.yml` *for a study*

Description

Write `metadata.yml` file with custom R code formatted to allow line breaks.

Usage

```
write_metadata(data, path, style_code = FALSE)
```

Arguments

`data`	`austraits` metadata object (a list)
`path`	Location where the metadata file is to be written to
`style_code`	Should the R code be styled?

Examples

```
## Not run:
f <- "data/Falster_2003/metadata.yml"
data <- read_metadata(f)
write_metadata(data, f)

## End(Not run)
```

write_metadata_dataset

Write the YAML representation of `metadata.yml` *for specified* `dataset_id` *to file* `data/dataset_id/metadata.yml`

Description

Write the YAML representation of `metadata.yml` for specified `dataset_id` to file `data/dataset_id/metadata.yml`

Usage

```
write_metadata_dataset(metadata, dataset_id)
```

Arguments

`metadata`	Metadata file
`dataset_id`	Identifier for a particular study in the database

Value

A yaml file

write_plaintext

Export AusTraits version as plain text

Description

Export AusTraits version as plain text

Usage

```
write_plaintext(austraits, path)
```

Arguments

`austraits`	AusTraits database object
`path`	Pathway to save file

Value

csv files of tibbles containing traits, locations, contexts, methods, excluded_data, taxonomic updates, taxa, contributors `write_yaml` *write yaml (from package yaml)*

Description

write yaml (from package yaml)

Index