Package ‘taxlist’

October 7, 2020

Version 0.2.0
Encoding UTF-8
Title Handling Taxonomic Lists
Depends R(>= 3.5.0)
Imports foreign, methods, stats, taxa, taxize, stringdist, utils, vegdata
Suggests ape, knitr, testthat, rmarkdown, roxygen2
LazyData true
Description Handling taxonomic lists through objects of class 'taxlist'.
   This package provides functions to import species lists from 'Turboveg'
   (<https://www.synbiosys.alterra.nl/turboveg>) and the possibility to create
   backups from resulting R-objects.
   Also quick displays are implemented as summary-methods.
License GPL (>= 2)
URL https://cran.r-project.org/package=taxlist,
    https://github.com/ropensci/taxlist,
    https://docs.ropensci.org/taxlist/
BugReports https://github.com/ropensci/taxlist/issues
Collate 'imports.R' 'internal.R' 'deprecated-functions.R'
   'replace_x.R' 'dissect_name.R' 'clean_strings.R'
   'taxlist-class.R' 'clean.R' 'as.list.R' 'taxon_views.R'
   'count_taxa.R' 'taxon_names.R' 'taxon_relations.R'
   'taxon_traits.R' 'levels.R' 'accepted_name.R' 'get_children.R'
   'merge_taxa.R' 'Extract.R' 'subset.R' 'backup_object.R'
   'summary.R' 'df2taxlist.R' 'tv2taxlist.R' 'tnrs.R'
   'tax2traits.R' 'match_names.R' 'print_name.R'
   'taxlist2taxmap.R' 'Easplist-data.R' 'taxlist-package.R'
VignetteBuilder knitr
RoxygenNote 7.1.1
NeedsCompilation no
Author  Miguel Alvarez [aut, cre] (<https://orcid.org/0000-0003-1500-1834>),
Zachary Foster [ctb] (<https://orcid.org/0000-0002-5075-0948>),
Sam Levin [rev],
Margaret Siple [rev]

Maintainer  Miguel Alvarez <kamapu78@gmail.com>

Repository  CRAN

Date/Publication  2020-10-07 14:00:05 UTC

R topics documented:

taxlist-package .......................... 3
accepted_name ................................ 3
as.list ..................................... 5
backup_object ................................ 6
clean ....................................... 8
clean_strings ................................ 9
count_taxa .................................. 10
Deprecated-functions ...................... 11
df2taxlist .................................. 11
dissect_name ................................ 13
Easplist-data ................................ 14
Extract ...................................... 15
get_children ................................ 16
levels ....................................... 17
match_names ................................ 18
merge_taxa ................................... 20
print_name ................................... 21
replace_x ................................... 23
subset ....................................... 25
summary ..................................... 26
tax2traits ................................... 28
taxlist-class ................................ 29
taxlist2taxmap ................................ 30
taxon_names ................................ 31
taxon_relations .............................. 33
taxon_traits ................................ 35
taxon_views ................................... 36
tns ........................................... 38
tv2taxlist ................................... 39

Index 40
taxlist-package  

**taxlist-package**  

**taxlist: Handling taxonomic lists.**

---

**Description**

The class `taxlist` is defined in this package using the S4 language. The main task of `taxlist` objects is to keep the taxonomic coherence of information included in taxonomic lists and to implement functions (methods) for a proper data handling. Objects of class `taxlist` can be included in further objects, for instance in biodiversity records as done in the package `vegetable`.

**Details**

The class `taxlist` is defined in this package using the S4 language. The main task of `taxlist` objects is to keep the taxonomic coherence of information included in taxonomic lists and to implement functions (methods) for a proper data handling. Objects of class `taxlist` can be included in further objects, for instance in biodiversity records as done in the package `vegetable`.

For a more detailed description of this package, see Alvarez & Luebert (2018).

**Author(s)**

Miguel Alvarez <kamapu78@gmail.com>

**References**


---

**accepted_name**  

Manage accepted names, synonyms and basionyms

---

**Description**

Taxon usage names for a taxon concept can be divided into three categories: accepted names, basionyms and synonyms. Each single taxon concept may at least have an accepted name, while basionym and synonyms are optional. The functions `accepted_name`, `basionym` and `synonyms` can be used either to display the respective usage names or to set usage names in one of those categories.

**Usage**

accepted_name(taxlist, ConceptID, ...)

```r
## S4 method for signature 'taxlist,numeric'
accepted_name(taxlist, ConceptID, show_traits = FALSE, ...)

## S4 method for signature 'taxlist,missing'
```
accepted_name(taxlist, ConceptID, ...) 

accepted_name(taxlist, ConceptID) <- value

## S4 replacement method for signature 'taxlist,numeric,numeric'
accepted_name(taxlist, ConceptID) <- value

synonyms(taxlist, ConceptID, ...)

## S4 method for signature 'taxlist,numeric'
synonyms(taxlist, ConceptID, ...)

## S4 method for signature 'taxlist,missing'
synonyms(taxlist, ConceptID, ...)

basionym(taxlist, ConceptID, ...)

## S4 method for signature 'taxlist,numeric'
basionym(taxlist, ConceptID, ...)

## S4 method for signature 'taxlist,missing'
basionym(taxlist, ConceptID, ...)

basionym(taxlist, ConceptID) <- value

## S4 replacement method for signature 'taxlist,numeric,numeric'
basionym(taxlist, ConceptID) <- value

Arguments

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>taxlist</td>
<td>An object of class taxlist.</td>
</tr>
<tr>
<td>ConceptID</td>
<td>Integer containing concept IDs where to request or set names for one category.</td>
</tr>
<tr>
<td>...</td>
<td>Further arguments passed among methods.</td>
</tr>
<tr>
<td>show_traits</td>
<td>Logical value, whether traits should be included in the output of accepted_name or not.</td>
</tr>
<tr>
<td>value</td>
<td>Integer containing usage IDs to be set to the respective category in the respective taxon concept.</td>
</tr>
</tbody>
</table>

Details

The function accepted_name retrieves the accepted names for the indicated taxon concepts or for the whole taxlist object. By using show_traits=TRUE, the respective taxon traits will be displayed as well, providing an overview of taxa included in the object. The replacement method for this function will set the respective usage name IDs as accepted names for the respective taxon concept, provided that these names are already set as synonyms in the respective concepts.

The function synonyms is working in a similar way as accepted_name, but this function does not include taxon traits in the output and there is no replacing method for synonyms. Alternatives for
inserting new synonyms into a taxon concept are either moving synonyms from other taxa by using `change_concept<-` or inserting new names in the object by using `add_synonym()`.

The function `basionym` is retrieving and setting basionyms in the respective taxon concepts similarly to `accepted_name`, but this function does not retrieve any information on taxon traits, either.

**Value**

Most of the methods return information in data frames, while replacement methods do it as `taxlist` objects.

**Author(s)**

Miguel Alvarez <kamapu78@gmail.com>

**See Also**

`add_synonym()` `change_concept<-` 

**Examples**

```r
## Set a different accepted name for Cyclosorus interruptus
summary(Easplist, "Cyclosorus interruptus")
accepted_name(Easplist, 50074) <- 53097
summary(Easplist, 50074)

## Inserting a new name first
summary(Easplist, "Basella alba")
Easplist <- add_synonym(taxlist=Easplist, ConceptID=68,
    TaxonName="Basella cordifolia", AuthorName="Lam.")
summary(Easplist, 68)
accepted_name(Easplist, 68) <- 56139
summary(Easplist, 68)
```

---

**as.list**

*Coerce an S4 object to a list.*

**Description**

Coercion of S4 objects to lists can be applied to explore their content, avoiding errors caused by their validation.

**Usage**

```r
S4_to_list(x)
```

```r
## S4 method for signature 'taxlist'
as.list(x, ...)
```
backup_object

Arguments

x An object of class taxlist or any S4 class.

... further arguments passed to or from other methods.

Details

The function S4_to_list transforms any S4 object to a list setting slots to elements of the list and it is running internally in the method as.list for taxlist objects.

Value

An object of class list.

Author(s)

Miguel Alvarez <kamapu78@gmail.com>

Examples

Easplist <- as.list(Easplist)
class(Easplist)

backup_object Make and load backups of R objects

Description

When work with data becomes risky, the best practice is to produce backup files. The function of backup_object is a wrapper of save(), adding a time stamp and a suffix to the name of the resulting file (an R image file with extension *.rda). The function load_last is adapted to this style, loading the newest version to the session.

Usage

backup_object(
  ..., 
  objects = character(),
  file,
  stamp = TRUE,
  overwrite = FALSE
)

load_last(file, fext = ".rda")
backup_object

Arguments

... Names of the objects to be saved (either symbols or character strings).
objects A character vector indicating the names of objects to be included in the backup file.
file A character value indicating the name of the backup file, without the extension.
stamp A logical value indicating whether time should be stamped in the backup name or not.
overwrite A logical value indicating whether existing files must be overwritten or not.
fdet A character value indicating the file extension (including the dot symbol).

Details

In both functions the argument file may include either the path relative to the working directory or the absolute path to the file, excluding stamps and extension. For overwrite=FALSE (the default), a numeric suffix will be added to the backup's name, if another backup was produced at the same day. For overwrite=TRUE no suffix will be included in the file and existing files will be overwritten.

The function load_last() will load the newest version among backups stored in the same folder, provided that the backup name includes a time stamp.

Value

An R image with extension *.rda.

Author(s)

Miguel Alvarez <kamapu78@gmail.com>

See Also

save load.

Examples

## Not run:
## A subset with Pseudognaphalium and relatives
Pseudognaphalium <- subset(x=Easplist, subset=grepl("Pseudognaphalium", TaxonName), slot="names")
Pseudognaphalium <- get_parents(Easplist, Pseudognaphalium)

## Create a backup with date stamp
backup_object(Pseudognaphalium, file="Pseudonaphalium")

## The same
backup_object(objects="Pseudognaphalium", file="Pseudonaphalium")

## To load the last backup into a session
load_last("Pseudognaphalium")
Manipulation of slots may generate orphaned entries in `taxlist` objects. The function `clean` deletes such entries and restores the consistency of the objects.

### Usage

```r
clean(object, ...) # output not shown
```

### Arguments

- `object` A `taxlist` object.
- `...` Further arguments passed from or to other methods.
- `times` An integer indicating how many times the cleaning should be repeated.

### Details

Cleaning of objects will follow the deletion of orphaned names, orphaned taxon trait entries, and orphaned parent entries.

### Value

A clean `taxlist` object.

### Author(s)

Miguel Alvarez.

### Examples

```r
# Direct manipulation of slot taxonRelations generates an invalid object
Easplist@taxonRelations <- Easplist@taxonRelations[1:5,]
# Not run:
summary(Easplist)
# End(Not run)
```
## Now apply cleaning

Easplist <- clean(Easplist)
summary(Easplist)

---

**clean_strings**

_Cleaning character strings._

### Description

Multiple, leading and trailing white spaces as well as wrong encodings may cause serious problems in information dealing with taxonomic names. The function `clean_strings` get rid of them.

### Usage

```r
clean_strings(x, ...)  
```

```r
## S4 method for signature 'character'
clean_strings(x, from = "utf8", to = "utf8", ...)
```

```r
## S4 method for signature 'factor'
clean_strings(x, from = "utf8", to = "utf8", ...)
```

```r
## S4 method for signature 'data.frame'
clean_strings(x, from = "utf8", to = "utf8", ...)
```

### Arguments

- **x**
  - Object to be cleaned.
- **...**
  - Further arguments passed among methods (not yet in use).
- **from, to**
  - Arguments passed to `iconv()`.

### Details

This function automatically deletes leading, trailing and multiple white spaces, either in strings (method character), levels (method factor) or in single columns (method data.frame).

### Value

The same as input `x`.

### Author(s)

Miguel Alvarez.
Examples

```r
library(taxlist)
clean_strings(" Cyperus papyrus L. ")
```

## count_taxa

Count taxa within a taxlist object.

Description

Counting number of taxa within `taxlist` objects or character vectors containing taxon names.

Usage

```r
count_taxa(object, data, ...)  
## S4 method for signature 'character,missing'
count_taxa(object, na.rm = TRUE, ...)  
## S4 method for signature 'factor,missing'
count_taxa(object, na.rm = TRUE, ...)  
## S4 method for signature 'taxlist,missing'
count_taxa(object, level, ...)  
## S4 method for signature 'formula,taxlist'
count_taxa(object, data, include_na = FALSE, suffix = "_count", ...)
```

Arguments

- `object` An object containing a taxonomic list or a formula.
- `data` An object of class `taxlist` in the `formula` method.
- `...` further arguments passed among methods.
- `na.rm` Logical value, whether NAs have to be removed from the input vector or not.
- `level` Character value indicating the taxonomic rank of counted taxa.
- `include_na` Logical value indicating whether NA values in a taxon trait should be considered for counting taxa or just ignored (only used in `formula` method).
- `suffix` Character value used as suffix for the counted rank in the output data frame (only used in `formula` method).

Details

This function is written by convenience in order to reduce code for counting taxa within `taxlist` objects and it is just a wrapper of `length()`. 
**Value**

An integer with the number of taxa.

**Author(s)**

Miguel Alvarez <kamapu78@gmail.com>

**Examples**

```r
## factor method
count_taxa(iris$Species)

## taxlist method
count_taxa(Easplist)
count_taxa(Easplist, level="species")

## using a formula
count_taxa(~ lf_behn_2018, Easplist)
```

---

**Deprecated functions**

Most of those functions have been replaced by alternative `update` ones.

**Usage**

- `add_parent()`
- `add_trait()`
- `add_level()`
- `replace_view()`

---

**df2taxlist**

Convert data frames into taxlist objects

**Description**

Taxon lists may be provided in data frame format, which will be converted to a `taxlist` object.
df2taxlist

Usage

df2taxlist(x, AcceptedName, ...)

## S4 method for signature 'data.frame,logical'
df2taxlist(x, AcceptedName, levels, ...)

## S4 method for signature 'data.frame,missing'
df2taxlist(x, AcceptedName, ...)

## S4 method for signature 'character,missing'
df2taxlist(x, AcceptedName, ...)

Arguments

- **x**: A data frame or a character vector with taxon names.
- **AcceptedName**: A logical vector indicating accepted names with value `TRUE`.
- **...**: Additional vectors to be added as columns in slot `taxonNames`.
- **levels**: A vector with the names of the taxonomic ranks. This argument is passed to `levels()`.

Details

In the method `data.frame`, the input data frame must have following columns:

- **TaxonUsageID**: Numeric code for the name.
- **TaxonConceptID**: Numeric code for the concept.
- **TaxonName**: Full name (usage), excluding author name.
- **AuthorName**: Author of the combination (taxon name).

If the argument `AcceptedName` is missing, all names will be assumed as accepted names. In the alternative character method, author names have to be added as additional vectors.

Be aware that the resulting object misses any information on taxon views, basionyms, parent concepts, hierarchical levels and taxon traits. All those elements can be added *a posteriori* by further functions provided in this package.

Value

A `taxlist` object.

Author(s)

Miguel Alvarez <kamapu78@gmail.com>.
**Examples**

```r
## Read the table with names of Cyperus species
Cyperus <- read.csv(file=file.path(path.package("taxlist"), "cyperus", 
    "names.csv"), stringsAsFactors=FALSE)
head(Cyperus)

## Convert to 'taxlist' object
Cyperus <- df2taxlist(Cyperus, AcceptedName != Cyperus$SYNONYM)
summary(Cyperus)

## Create a 'taxlist' object from character vectors
Plants <- df2taxlist(c("Triticum aestivum", "Zea mays"), AuthorName="L.")
summary(Plants, "all")
```

---

**dissect_name**

**Dissect Scientific Names into their Elements**

**Description**

Depending the degree of resolution and specific roles of nomenclature, strings containing taxon usage names (scientific names) are constructed with different parts. A string with names can be consequently split into those elements, meanwhile the number of elements will suggest the taxonomic ranks.

**Usage**

```r
dissect_name(x, split = " ", fixed = TRUE, ...)
```

**Arguments**

- `x`: A character vector containing taxon names.
- `split, fixed, ...`: Arguments passed to `strsplit()`.

**Details**

This function is using `strsplit()` for splitting names. Single spaces will be used to dissect names but it can be changed in the value of argument `split`. The number of columns in the resulting matrix will depend on the longest polynomial string.

**Value**

A character matrix with as many rows as names in the input vector.

**Author(s)**

Miguel Alvarez <kamapu78@gmail.com>
See Also

strsplit()

Examples

Easplist <- subset(x=Easplist, subset=Level == "variety", slot="relations")
Easplist <- accepted_name(Easplist)[c(1:10),"TaxonName"]
dissect_name(Easplist)

---

**Easplist-data**

*List of vascular plants from East Africa*

Description

Example of an incomplete taxonomic list including taxa recorded in East Africa.

Usage

Easplist

Format

An object of class `taxlist`.

Details

This list is a subset of the taxonomic list implemented in the database SWEA-Dataveg. Since this list is being complemented regarding stored vegetation plots, it is an incomplete list.

Source

African Plant Database, SWEA-Dataveg.

Examples

summary(Easlist)
Extract or Replace Parts of taxlist Objects

Description

Quick access to slots taxonTraits and taxonRelations within taxlist objects.

Usage

```r
## S4 method for signature 'taxlist'
x[i, j, drop = FALSE]
```

```r
## S4 method for signature 'taxlist'
x$name
```

Arguments

- `x`: Object of class taxlist.
- `i`: Integer or logical vector used as index for access to taxon concepts, referring to the rows in slot 'taxonRelations'. These indices can be used to produce a object with a subset of taxon concepts. It is not recommended to use character values for this index.
- `j`: Integer, logical or character vector used as index for access to variables in slot 'taxonTraits'. These indices can be used to reduce the number of variables in the mentioned slot.
- `drop`: A logical value passed to `Extract`.
- `name`: A symbol or character value for the method $, corresponding to a variable either at slot 'taxonTraits' or slot 'taxonRelations'.

Value

The method $ retrieves a vector, while [] retrieves a subset of the input taxlist object.

Author(s)

Miguel Alvarez <kamapu78@gmail.com>.

See Also

- `taxlist`
- `subset`
get_children

Retrieve children or parents of taxon concepts

Description

Retrieve all children or all parents of a queried taxon concept.

Usage

get_children(taxlist, ConceptID, ...)

## S4 method for signature 'taxlist,numeric'
get_children(taxlist, ConceptID, ...)

## S4 method for signature 'taxlist,taxlist'
get_children(taxlist, ConceptID, ...)

get_parents(taxlist, ConceptID, ...)

## S4 method for signature 'taxlist,numeric'
get_parents(taxlist, ConceptID, ...)

## S4 method for signature 'taxlist,taxlist'
get_parents(taxlist, ConceptID, ...)

Arguments

taxlist A taxlist object.
ConceptID Concept IDs for selecting parents or children or a subset of taxlist.
... Further arguments passed among methods.

Details

This function produces subsets of taxlist objects including all children or parents of queried taxon concepts. Multiple concepts can be queried in these function. The argument ConceptID can be a vector of concept IDs or a subset of the input taxlist object.

Value

A taxlist object with a subset including requested concepts with children or parents.
levels

Set and retrieves hierarchical levels

Description

Taxonomic hierarchies can be set as levels in taxlist objects, ordered from lower to higher levels. Add taxonomic levels for specific taxon concepts in a taxlist object. Also changes in concept circumscription may implicate changes in its taxonomic hierarchy.

Usage

## S4 method for signature 'taxlist'
levels(x)

## S4 replacement method for signature 'taxlist'
levels(x) <- value

Arguments

x A taxlist object.
value A character vector with replacement values for levels of x.
Details

Taxonomic levels will be handled as factors in the taxlist objects. Those levels are useful for creating subsets of related groups (e.g. by functions \texttt{get\_children()}} or \texttt{get\_parents()}).

Levels in combination to parent-child relationships will be further used for checking consistency of taxonomic lists.

A replacement method of the form \texttt{levels(x) <- value} it is also implemented.

Value

A character vector or a taxlist object with added or modified taxonomic levels.

Author(s)

Miguel Alvarez <kamapu78@gmail.com>

Examples

```r
## Get levels of species list
taxlist::levels(Easplist)

## Add aggregate as new taxonomic level
levels(Easplist) <- c("form", "variety", "subspecies", "species",
"complex", "aggregate", "genus", "family")
summary(Easplist)
```

Description

Names provided in a character vector will be compared with names stored in slot \texttt{taxonNames} within an object of class \texttt{taxlist} by using the function \texttt{stringsim()}.

Usage

```r
match_names(x, object, ...)

## S4 method for signature 'character,character'
match_names(x, object, best = 5, clean = TRUE, decreasing = TRUE, ...)

## S4 method for signature 'character,taxlist'
match_names(
  x,
  object,
  clean = TRUE,
  output = "data.frame",
```
match_names

best = 5,
show_concepts = FALSE,
accepted_only = FALSE,
method = "lcs",
decreasing,
... )

Arguments

x A character vector with names to be compared.

object An object of class taxlist to be compared with.

best Integer value indicating how many from the best matches have to be displayed (only working for output="list").

clean Logical value, whether leading, tailing and double blanks should be deleted from x.

decreasing Logical value indicating whether retrieved names should be sorted by decreasing or increasing similarity value. In the character method, the sorting corresponds to similarities between the queried value and the reference vector (argument object). In the taxlist method using output="data.frame", the order corresponds to the similarity of the best match (by default, no sorting is done). This argument is passed to order().

output Character value indicating the type of output. Alternative values are "list" (taxon concepts ID’s sorted by similarity for each queried name) or "data.frame" (a table including the best match for every queried name).

show_concepts Logical value, whether respective concepts should be displayed in output or not.

accepted_only Logical value, whether only accepted names should be matched or all usage names (including synonyms).

method, ... Further arguments passed to stringsim().

Author(s)

Miguel Alvarez <kamapu78@gmail.com>

See Also

stringsim()

Examples

## Names to be compared
species <- c("Cyperus papyrus", "Typha australis", "Luke skywalker")

## Comparing character vectors
match_names("Cyperus paper", species)

## Retrieve taxon usage names
match_names(species, Easplist)
merge_taxa

## Display accepted names in output
match_names(x=species, object=Easplist, show_concepts=TRUE)

### Description
Merge taxon concepts from a taxlist object into single ones.

### Usage

merge_taxa(object, concepts, level, ...)

- **object**, taxlist: Object of class taxlist.
- **concepts**: Numeric (integer) vector including taxon concepts to be merged.
- **level**: Character vector indicating the lowest level for merging.
- **...**: Further arguments to be passed to or from other methods.
- **print_output**: Logical value indicating whether the merged concept should be displayed in the console.
- **UsageID**: Numeric vector with taxon usage IDs to be changed from concept.
- **value**: Numeric vector with taxon concept IDs to be assigned to the names.

### Details
Taxon concepts indicated in argument concepts will be merged into a single concept. The new concept inherits the ID and respective attributes from slots taxonRelations and taxonTraits from the first taxon concept indicated in argument concepts.

For convenience the resulting concept can be displayed by setting print_output=TRUE but only when using argument concepts.
An alternative application of this function is implemented through the argument `level`, where all lower rank taxa will be merged to the indicated level or higher (if parent of merged taxa are at a higher rank).

Value

An object of class `taxlist`.

Author(s)

Miguel Alvarez <kamapu78@gmail.com>

Examples

```r
## Merge Cyperus papyrus and Cyperus dives
summary(Easplist, c(206, 197))
Easplist <- merge_taxa(object=Easplist, concepts=c(206, 197),
                      print_output=TRUE)

## Move the name Typha aethiopica to concept 573 (T. latifolia)
change_concept(Easplist, 53130) <- 573
summary(Easplist, c(50105, 573))
```

---

print_name

Format usage names for publications

Description

When writing on bio-diversity, usage names could be automatically inserted in documents including the typical italic format for different elements of a scientific name. The function `print_name` can be applied either in markdown documents or for graphics.

Usage

```r
print_name(object, id, ...)

## S4 method for signature 'taxlist,numeric'
print_name(
  object,
  id,
  concept = TRUE,
  second_mention = FALSE,
  include_author = TRUE,
  secundum,
  style = "markdown",
  ...
)
```
print_name

Arguments

- **object**: An object of class `taxlist`.
- **id**: Integer containing either a concept or a name ID.
- **...**: Further arguments passed among methods.
- **concept**: Logical value, whether id corresponds to a concept ID or a taxon usage name ID.
- **second_mention**: Logical value, whether the genus name should be abbreviated or not.
- **include_author**: Logical value, whether authors of the name should be mentioned or not.
- **secundum**: Character value indicating the column in slot `taxonViews` that will be mentioned as `secundum` (according to).
- **style**: Character value indicating the alternative format for italics (at the moment only markdown and html implemented).

Details

In **Rmarkdown** documents use *Cyperus papyrus* L. for inserting a formatted a species name.

Value

A character value including format to italic font.

Author(s)

Miguel Alvarez <kamapu78@gmail.com>

See Also

`ape::mixedFontLabel()`.

Examples

```r
summary(Easplist, 363, secundum="secundum")

## Empty plot
plot(x=NA, xlim=c(0,5), ylim=c(7,1), bty="n", xaxt="n", xlab="", ylab="options")

## Accepted name with author
text(x=0, y=1, labels=print_name(Easplist, 363, style="expression"), pos=4)

## Including taxon view
text(x=0, y=2, labels=print_name(Easplist, 363, style="expression", secundum="secundum"), pos=4)

## Second mention in text
text(x=0, y=3, labels=print_name(Easplist, 363, style="expression", second_mention=TRUE), pos=4)
```
replace_x

## Using synonym
```r
text(x=0, y=4, labels=print_name(Easplist, 50037, style="expression", concept=FALSE), pos=4)
```

## Markdown style
```r
text(0, 5, labels=print_name(Easplist, 363, style="markdown"), pos=4)
```

## HTML style
```r
text(0, 6, labels=print_name(Easplist, 363, style="html"), pos=4)
```

## LaTeX style for knitr
```r
text(x=0, y=7, labels=print_name(Easplist, 363, style="knitr"), pos=4)
```

---

**replace_x**  
*Data manipulation.*

### Description

This is a series of functions designed for a fast coding of replacements both, as internal functions and in workflows dealing with information stored in vectors and data frames. Such functions are especially useful when handling with functional traits stored in taxlist objects.

*replace_x()* is used to exchange values in vectors. *replace_idx()* changes values in vectors by matching indices or conditions. The function *replace_na()* works in the same way as *replace_idx()* but will only insert values in empty elements (NAs).

The function *insert_rows()* will add rows and columns at the same time. This function will be used when a new table is appended to another but sharing only part of the columns.

### Usage

```r
replace_x(x, old, new)
replace_idx(x, idx1, idx2, new)
replace_na(x, idx1, idx2, new)
insert_rows(x, y)
```

### Arguments

- **x**  
  A vector to be modified. In the case of *insert_rows()* , *x* is a data frame.
- **old**  
  A vector with values to be replaced by *replace_x()* in a vector.
- **new**  
  A vector containing values to be inserted, either comparing values or using indices.
- **idx1, idx2**  
  Indices applied for value replacements to match *x* with *new*, respectively.
- **y**  
  A data frame including rows (and columns) to be inserted in *x*.
replace_x

Value
A vector or data frame with the modified values.

Author(s)
Miguel Alvarez.

Examples

```r
## Replace values in vector
replace_x(x=letters, old=c("b", "p", "f"), new=c("bee", "pork", "fungus"))
```

```r
## Replace values using indices
replace_idx(x=letters, idx1=1:length(letters), idx2=c(2, 7, 17),
            new=c("second", "seventh", "seventeenth"))
```

```r
## Replace values if they are NAs
letters[2] <- NA
replace_na(x=letters, idx1=1:length(letters), idx2=c(1:3),
           new=c("alpha", "beta", "zeta"))
```

```r
## The same applications but this time for functional traits
summary(as.factor(Easplist$lf_behn_2018))
```

```r
# Merge annuals
Easplist@taxonTraits$lifeform <- replace_x(
    x=Easplist@taxonTraits$lf_behn_2018,
    old=c("obligate_annual", "facultative_annual"),
    new=c("annual", "annual"))
summary(as.factor(Easplist$lifeform))
```

```r
# The same effect
Easplist@taxonTraits$lifeform <- replace_idx(
    x=Easplist@taxonTraits$lf_behn_2018,
    idx1=grepl("annual", Easplist@taxonTraits$lf_behn_2018),
    idx2=TRUE,
    new="annual")
summary(as.factor(Easplist$lifeform))
```

```r
## Merge data frames including new columns
data(iris)
iris$Species <- paste(iris$Species)
new_iris <- data.frame(Species=rep("humilis", 2), Height=c(15,20),
                        stringsAsFactors=FALSE)
insert_rows(iris, new_iris)
```
Description

Subset of taxlist objects will be done applying either logical operations or pattern matchings. Subsets can be referred to information contained either in the slot taxonNames, taxonRelations or taxonTraits.

Usage

```r
## S4 method for signature 'taxlist'
subset(
x, subset, slot = "names", keep_children = FALSE, keep_parents = FALSE, ...
)
```

Arguments

- `x` Object of class taxlist.
- `subset` Logical vector or logical operation to apply as subset.
- `slot` Character value indicating the slot to be used for the subset.
- `keep_children` Logical value applied to hierarchical structures.
- `keep_parents` Logical value applied to hierarchical structures.
- `...` Further arguments to be passed to or from other methods.

Details

The argument subset will be applied to the slot specified in argument slot. This argument also allows partial matchings.

Arguments keep_children and keep_parents are applied to objects including parent-child relationships. When those arguments are set as FALSE (the default), children or parents of selected taxon concepts will not be included in the subset.

Be aware that subset() won’t work properly inside of function definitions.

Value

An object of class taxlist.

Author(s)

Miguel Alvarez <kamapu78@gmail.com>
Examples

```r
Easplist <- subset(x=Easplist, subset=lf_behn_2018 == "reed_plant", slot="traits")
summary(Easplist)

summary(as.factor(Easplist$lf_behn_2018))
```

**summary**

*Print overviews for taxlist Objects and their content*

**Description**

A method to display either an overview of the content of `taxlist` objects or an overview of selected taxa.

**Usage**

```r
## S4 method for signature 'taxlist'
summary(
  object, 
  ConceptID, 
  units = "Kb", 
  check_validity = TRUE, 
  display = "both", 
  maxsum = 5, 
  secundum = NULL, 
  ...
)
```

```r
## S4 method for signature 'taxlist'
show(object)
```

```r
## S4 method for signature 'taxlist'
print(x, ...)
```

**Arguments**

- **object, x** A `taxlist` object.
- **ConceptID** IDs of concepts to be displayed in the summary.
- **units** Character value indicating the units shown in the object’s allocated space.
- **check_validity** Logical value indicating whether the validity of `object` should be checked or not.
- **display** Character value indicating the field to be displayed (see details).
- **maxsum** Integer indicating the maximum number of displayed taxa.
secundum A character value indicating the column from slot taxonViews to be displayed in the summary.

... Further arguments passed to or from another methods.

Details

A general overview indicating number of names, concepts and taxon views included in taxlist objects. If argument ConceptID is a vector with concept IDs or names to be matched by grepl(), then a display of all names included in each concept will be produced. Alternative you can use taxon="all" in order to get the listing of names for all concepts included in the object (truncated to the input number of maxsum).

For summaries applied to concepts, there are three alternative displays of names using the argument display. Use display="name" to show the value TaxonName, display="author" to show the value AuthorName or display="both" to show both values. Such values are taken from slot taxonNames.

For big objects it will be recommended to set units="Mb" (see also object.size() for further alternatives).

Author(s)

Miguel Alvarez <kamapu78@gmail.com>

See Also

taxlist

Examples

```r
## summary of the object
summary(Easplist, units="Mb")

## the same output
summary(Easplist)
show(Easplist)
print(Easplist)
Easplist

## summary for two taxa
summary(Easplist, c(51128,51140))

## summary for a name
summary(Easplist, "Acmella")

## summary for the first 10 taxa
summary(object=Easplist, ConceptID="all", maxsum=10)
```
tax2traits

Set taxonomic information as taxon traits

Description

Taxonomic classification can be included in taxlist objects within the information provided at slot taxonRelations. Nevertheless, for statistical analyses it may be more convenient to insert such information in the slot taxonTraits.

Usage

tax2traits(object, ...)

## S4 method for signature 'taxlist'
tax2traits(object, get_names = FALSE, ...)

Arguments

- object: An object of class taxlist.
- ...: Further arguments to be passed among methods.
- get_names: Logical value indicating whether taxon names should be retrieved instead of taxon IDs.

Details

This function can only be applied to objects containing parent-child relationships and information on taxonomic levels.

Value

An object of class taxlist with taxonomy added as traits.

Author(s)

Miguel Alvarez <kamapu78@gmail.com>.

Examples

## Family Acanthaceae with children
Acanthaceae <- subset(x=Easplist, subset=TaxonName == "Acanthaceae",
slot="names", keep_children=TRUE)
summary(Acanthaceae)

## Insert taxonomy to taxon traits
Acanthaceae <- tax2traits(Acanthaceae, get_names=TRUE)
head(taxon_traits(Acanthaceae))
taxlist-class

An S4 class to represent taxonomic lists.

Description

Class for taxonomic lists including synonyms, hierarchical ranks, parent-child relationships, taxon views and taxon traits.

Note that each taxon becomes an identifier, represented by the column `TaxonConceptID` in the slot `taxonRelations`, analogous to a primary key in a relational database. This identifier is restricted to an integer in `taxlist` and is specific for the object.

In the same way, each taxon usage name has an identifier in the column `TaxonUsageID`, slot `taxonNames`. The column `ViewID` in slot `taxonViews` is the identifier of the taxon view.

Slots

taxonNames (data.frame) Table of taxon usage names (accepted names and synonyms).
taxonRelations (data.frame) Relations between concepts, accepted names, basionyms, parents and hierarchical level.
taxonTraits Table of taxon traits.
taxonViews References used to determine the respective concept circumscription.

Author(s)

Miguel Alvarez

References


Examples

library(taxlist)

showClass("taxlist")

## Create an empty object
Splist <- new("taxlist")
taxlist2taxmap

Conversion among taxlist and taxmap objects

Description

Exchange of data between the packages taxlist and taxa.
This function should be used for the exchange of data between the packages taxlist-package and taxa.

Usage

taxlist2taxmap(taxlist, ...)

## S4 method for signature 'taxlist'
taxlist2taxmap(taxlist, ...)

taxmap2taxlist(taxmap, relations, traits, synonyms, views, reindex = FALSE)

Arguments

taxlist Input object of class taxlist.
...
Additional arguments passed among methods.
taxmap Input object of class taxmap.
relations, traits, synonyms, views
Character values indicating the names of data frames in the taxmap object at data, which should be used for the slots taxonRelations, taxonTraits, taxonNames, and taxonViews, respectively.
reindex Logical value indicating whether taxon IDs should be assigned anew or not.

Value

Depending on the applied function, either a taxlist or a Taxmap object.

Author(s)

Miguel Alvarez (<kamapu78@gmail.com>) and Zachary Foster (<zacharyfoster1989@gmail.com>).

Examples

## Subset Easplist
Cyperus <- subset(Easplist, grepl("Cyperus", TaxonName))

## Convert to taxmap
Cyperus2 <- taxlist2taxmap(Cyperus)
Cyperus2

## Convert it back to taxlist
Cyperus2 <- taxmap2taxlist(taxmap=Cyperus2, traits="traits", views="views", synonyms="synonyms")
Cyperus2

taxon_names

Handle information on taxon usage names.

Description

The slot taxonNames in taxlist objects contains taxon usage names for the respective taxon. These functions assist on the access and modification of entries for names.

Usage

taxon_names(taxlist, ...)

## S4 method for signature 'taxlist'
taxon_names(taxlist, ...)
taxon_names(taxlist) <- value

## S4 replacement method for signature 'taxlist,data.frame'
taxon_names(taxlist) <- value

add_synonym(taxlist, ConceptID, ...)

## S4 method for signature 'taxlist'
add_synonym(taxlist, ConceptID, TaxonName, AuthorName, ...)

update_name(taxlist, UsageID, ...)

## S4 method for signature 'taxlist,numeric'
update_name(taxlist, UsageID, ...)

delete_name(taxlist, UsageID, ...)

## S4 method for signature 'taxlist,numeric'
delete_name(taxlist, UsageID, ...)

Arguments

taxlist A taxlist object to be modified.
...

Further arguments passed among methods. In update_name are vectors including the variables to be updated for the respective taxon usage ID.

value A data frame used as new slot taxonNames in taxlist.

ConceptID Numeric vector indicating the concept ID to which the synonyms will be added.

taxon_names

TaxonName, AuthorName
Character values used for the new names (synonyms).

UsageID
Numeric vector indicating the taxon usage IDs to be updated.

Details
The replacement method taxon_names<- is a quick alternative to include names in empty taxlist objects.

The function add_synonym() works only for adding names to existing taxon concepts. For adding new taxon concepts as well you should use add_concept().

Value
A data frame or, in the case of the replacement method, a taxlist object with modified slot taxonNames.

Author(s)
Miguel Alvarez <kamapu78@gmail.com>

See Also
taxlist

Examples

## Display of slot 'taxonNames'
Euclea <- subset(Easplist, charmatch("Euclea", TaxonName), 
   slot="names", keep_children=TRUE)
Euclea
taxon_names(Euclea)

## Insert a synonym to Diospyros scabra
summary(Easplist, "Diospyros scabra")
Easplist <- add_synonym(taxlist=Easplist, ConceptID=51793, 
   TaxonName="Maba scabra", AuthorName="Chiov.")
summary(Easplist, "Diospyros scabra")

## Delete a synonym of Launaea cornuta
summary(Easplist, "Launaea cornuta")
Easplist <- delete_name(Easplist, 53821)
summary(Easplist, "Launaea cornuta")

## Hypothetical correction in author name in Launaea cornuta
Easplist <- update_name(taxlist=Easplist, UsageID=355, AuthorName="L.")
summary(Easplist, "Launaea cornuta")
taxon_relations

Retrieve or replace slot taxonRelations in taxlist objects

Description

Retrieve the content of slot taxonRelations from a taxlist object or replace it by a new data frame.

Usage

taxon_relations(taxlist, ...)

```r
## S4 method for signature 'taxlist'
taxon_relations(taxlist, ...)
```

taxon_relations(taxlist) <- value

```r
## S4 replacement method for signature 'taxlist, data.frame'
taxon_relations(taxlist, ...) <- value
```

add_concept(taxlist, TaxonName, ...)

```r
## S4 method for signature 'taxlist, character'
add_concept(taxlist, TaxonName, Level, ...)
```

```r
## S4 method for signature 'taxlist, taxlist'
add_concept(taxlist, TaxonName, insert_view, ...)
```

update_concept(taxlist, ConceptID, ...)

```r
## S4 method for signature 'taxlist, numeric'
update_concept(taxlist, ConceptID, ...)
```

Arguments

taxlist A taxlist object.

... Further arguments passed among methods.

value A data.frame object to be set as slot taxonRelations.

TaxonName Character vector with the accepted name for the new taxon concepts.

Level Character vector indicating the level of the concept in the list.

insert_view A numeric (integer) vector, indicating the views to be inserted in taxlist or the value TRUE (see details).

ConceptID Concept IDs to be updated.
Details

The replacement method `taxon_relations<-` should be only used when constructing `taxlist` objects from an empty one (prototype).

New concepts should be first added to a `taxlist` object using their respective accepted names. Synonyms can be further provided using the function `add_synonym()`.

Additional named vectors can be provided to be included in slot `taxonNames`, in the cases where those variables already exist, otherwise they will be ignored.

It is recommended also to provide a concept view as `ViewID` (see `taxon_views()`). For adding a new view, use `add_view()`.

Value

An object of class `taxlist` with added names and concepts.

Author(s)

Miguel Alvarez <kamapu78@gmail.com>

See Also

taxlist

Examples

## Subset for the genus Euclea and display of slot 'taxonNames'
Euclea <- subset(x=Easplist, subset=charmatch("Euclea", TaxonName),
  slot="names")
Euclea <- get_children(Easplist, Euclea)

Euclea
taxon_relations(Euclea)

## Subset with family Ebenaceae and children
Ebenaceae <- subset(Easplist, charmatch("Ebenaceae", TaxonName))
Ebenaceae <- get_children(Easplist, Ebenaceae)

Ebenaceae
summary(object=Ebenaceae, ConceptID="all", maxsum=100)

## Adding a new concept
Ebenaceae <- add_concept(taxlist=Ebenaceae, TaxonName="Euclea acutifolia",
  AuthorName="E. Mey. ex A. DC.", Level="species", Parent=55707, ViewID=1)

## A summary again
Ebenaceae
summary(Ebenaceae, "all", maxsum=100)

## Display two Typha species
summary(Easplist, c("Typha domingensis","Typha latifolia"))
## Update a concept

```r
summary(Easplist, "Corchorus olitorius")
Easplist <- update_concept(taxlist=Easplist, ConceptID=155,
  Level="subspecies")
summary(Easplist, "Corchorus olitorius")
```

---

### Description

The slot `taxonTraits` in `taxlist` objects contains attributes of taxon concepts (e.g. functional traits). These functions are suitable for replacing, retrieving and appending trait information in taxonomic lists.

### Usage

```r
taxon_traits(taxlist, ...
```

```r
## S4 method for signature 'taxlist'
taxon_traits(taxlist, ...)
```

```r
taxon_traits(taxlist) <- value
```

```r
## S4 replacement method for signature 'taxlist,data.frame'
taxon_traits(taxlist) <- value
```

```r
update_trait(taxlist, ConceptID, ...)
```

```r
## S4 method for signature 'taxlist,numeric'
update_trait(taxlist, ConceptID, ...)
```

### Arguments

- `taxlist` A `taxlist` object.
- `...` Further arguments to be passed among methods.
- `value` Data frame to be set as slot `taxonTraits`.
- `ConceptID` A numeric vector with the respective taxon concept IDs.

### Details

Taxon traits are contained in a data frame at the slot `taxonTraits` in `taxlist` objects. To optimise space, this data frame contain only entries for those concepts with information, while taxa with no information are skipped from this table. Thus appending new variables may also have to include new rows in this slot, which is automatically carried out by this function.

The replacement method `taxon_traits<-` should be only used when constructing `taxlist` objects from an empty one.
Author(s)

Miguel Alvarez <kamapu78@gmail.com>

See Also
taxlist.

Examples

head(taxon_traits(Easplist))

## Updating traits for Launaea cornuta
summary(Easplist, "Launaea cornuta")
accepted_name(taxlist=Easplist, ConceptID=355, show_traits=TRUE)

# Update
Easplist <- update_trait(taxlist=Easplist, ConceptID=355,
  lf_benh_2018="annual")
accepted_name(taxlist=Easplist, ConceptID=355, show_traits=TRUE)

Description

Retrieve or replace slot taxonViews in an object of class taxlist

Usage

taxon_views(taxlist, ...)

## S4 method for signature 'taxlist'
taxon_views(taxlist, ...)
taxon_views(taxlist) <- value

## S4 replacement method for signature 'taxlist, data.frame'
taxon_views(taxlist) <- value

add_view(taxlist, ...)

## S4 method for signature 'taxlist'
add_view(taxlist, ...)
Arguments

- **taxlist**: A `taxlist` object.
- **value**: An object of class `data.frame` containing the references used to define the circumscription of taxon concepts included in `taxlist`.

Details

Taxon views indicate in `taxlist` objects the references determining the circumscription of the respective taxon concepts. When adding a new concept (see `add_concept()`), the respective reference may not yet occur in the input `taxlist` object.

The term taxon view was introduced by Zhong et al. (1996) and corresponds to the reference used for the definition of a concept.

This function retrieves the slot `taxonViews` from objects of the class `taxlist`.

The replacement method `taxon_views<-` replaces the whole content of slot `taxonViews` and it is only recommended to use when constructing a new `taxlist` object from an empty prototype.

Value

An object of class `taxlist` with added views.

Author(s)

Miguel Alvarez <kamapu78@gmail.com>

References


See Also

taxlist

Examples

```r
taxon_views(Easplist)

## Add a new view
Easplist <- add_view(taxlist=Easplist, secundum="Beentje et al. (1952)",
                     Title="Flora of Tropical East Africa",
                     URL="http://www.kew.org/science/directory/projects/FloraTropEAfrica.html")

taxon_views(Easplist)
```
Description

Methods of `taxize::tnrs()` for taxlist objects.

Usage

tnrs(query, ...)

## S4 method for signature 'character'
tnrs(query, ...)

## S4 method for signature 'taxlist'
tnrs(query, min_score = 0.8, source = "iPlant_TNRS", ...)

Arguments

query          Either a character vector or a taxlist object with names to search.
...            Further arguments passed to `taxize::tnrs()`.
min_score      Minimum value of score for considering accepted names as suggested by the output.
source         Source database.

Details

This function checks for matching of taxon names in taxlist objects with the Taxonomic Name Resolution Service (TNRS). Misspelled names as well as author names will be replaced in the new object and new accepted names will be inserted.

A method for character vectors is defined for the original function.

Value

A data frame or an object of class taxlist.

Author(s)

Miguel Alvarez <kamapu78@gmail.com>

See Also

taxize::tnrs()
**tv2taxlist**

Import species lists from Turboveg databases

**Description**

Importing species lists from Turboveg [https://www.synbiosys.alterra.nl/turboveg/](https://www.synbiosys.alterra.nl/turboveg/) databases into an object of class `taxlist`.

**Usage**

```r
tv2taxlist(taxlist, tv_home = tv.home())
```

**Arguments**

- `taxlist` The name of a species list in Turboveg as character value.
- `tv_home` Character value indicating the path to the main Turboveg folder.

**Details**

This function imports species lists using the function `read.dbf()`. When available, also taxon traits will be imported into the output object (usually the file `ecodbase.dbf`). During import of taxon traits, duplicated entries for a same concept will be discarded as well as entries for non-existing concepts.

By default `tv_home` will be set by the function `tv.home()` from the package `vegdata-package`.

By default, the name of the database will be set as concept view for all concepts included in the species list. If this is not correct, consider setting it manually by using the functions `taxon_views()` and `add_view()`.

**Value**

An object of class `taxlist`.

**Author(s)**

Miguel Alvarez <kamapu78@gmail.com>

**See Also**

- `taxlist`

**Examples**

```r
## Cyperus data set installed as Turboveg species list
Cyperus <- tv2taxlist(taxlist="cyperus",
                       tv_home=file.path(path.package("taxlist"), "tv_data"))

summary(Cyperus)
```
Index

* datasets
  * Easlist-data, 14
  [ (Extract), 15
  [, ,taxlist-method (Extract), 15
  $ (Extract), 15
  $,taxlist-method (Extract), 15
  accepted_name, 3
  accepted_name, taxlist, missing-method
      (accepted_name), 3
  accepted_name, taxlist, numeric-method
      (accepted_name), 3
  accepted_name<-, (accepted_name), 3
  accepted_name<-, taxlist, numeric, numeric-method
      (accepted_name), 3
  add_concept (taxon_relations), 33
  add_concept (), 32, 37
  add_concept, taxlist, character-method
      (taxon_relations), 33
  add_concept, taxlist, taxlist-method
      (taxon_relations), 33
  add_concept, taxlist, taxlist-method
      (taxon_relations), 33
  add_level (Deprecated-functions), 11
  add_parent (Deprecated-functions), 11
  add_synonym (taxon_names), 31
  add_synonym(), 5, 34
  add_synonym, taxlist-method
      (taxon_names), 31
  add_trait (Deprecated-functions), 11
  add_view (taxon_views), 36
  add_view(), 34, 39
  add_view, taxlist-method (taxon_views),
      36
  ape::mixedFontLabel(), 22
  as.list, 5
  as.list, taxlist-method (as.list), 5
  backup_object, 6
  basionym (accepted_name), 3
  basionym, taxlist, missing-method
      (accepted_name), 3
  basionym, taxlist, numeric-method
      (accepted_name), 3
  basionym<-, (accepted_name), 3
  basionym<-, taxlist, numeric, numeric-method
      (accepted_name), 3
  change_concept<-, 5
  change_concept<-, (merge_taxa), 20
  change_concept<-, taxlist-method
      (merge_taxa), 20
  clean, 8
  clean, taxlist-method (clean), 8
  clean_strings, 9
  clean_strings, character-method
      (clean_strings), 9
  clean_strings, data.frame-method
      (clean_strings), 9
  clean_strings, factor-method
      (clean_strings), 9
  count_taxa, 10
  count_taxa, character, missing-method
      (count_taxa), 10
  count_taxa, factor, missing-method
      (count_taxa), 10
  count_taxa, formula, taxlist-method
      (count_taxa), 10
  count_taxa, taxlist, missing-method
      (count_taxa), 10
  data.frame, 37
  delete_name (taxon_names), 31
  delete_name, taxlist, numeric-method
      (taxon_names), 31
  Deprecated-functions, 11
  df2taxlist, 11
  df2taxlist, character, missing-method
      (df2taxlist), 11
  df2taxlist, data.frame, logical-method
      (df2taxlist), 11
df2taxlist, data.frame, missing-method (df2taxlist), 11
dissect_name, 13
Easplist (Easplist-data), 14
Easplist-data, 14
Extract, 15, 15
get_children, 16
get_children(), 18
get_children, taxlist, numeric-method (get_children), 16
get_children, taxlist, taxlist-method (get_children), 16
get_parents (get_children), 16
get_parents(), 18
get_parents, taxlist, numeric-method (get_children), 16
get_parents, taxlist, taxlist-method (get_children), 16
grepl(), 27
iconv(), 9
insert_rows (replace_x), 23
length(), 10
levels, 17
levels(), 12
levels, taxlist-method (levels), 17
levels<-(levels), 17
levels<-, taxlist-method (levels), 17
list, 6
load, 7
load_last (backup_object), 6
match_names, 18
match_names, character, character-method (match_names), 18
match_names, character, taxlist-method (match_names), 18
merge_taxa, 20
merge_taxa, taxlist, missing, character-method (merge_taxa), 20
merge_taxa, taxlist, numeric, missing-method (merge_taxa), 20
object.size(), 27
order(), 19
print, taxlist-method (summary), 26
print_name, 21
print_name, taxlist, numeric-method (print_name), 21
read.dbf(), 39
replace_idx (replace_x), 23
replace_na (replace_x), 23
replace_view (Deprecated-functions), 11
replace_x, 23
S4_to_list (as.list), 5
save, 7
save(), 6
show, taxlist-method (summary), 26
stringsim(), 18, 19
strsplit(), 13, 14
subset, 15, 25
subset(), 25
subset, taxlist-method (subset), 25
summary, 26
summary, taxlist-method (summary), 26
synonyms (accepted_name), 3
synonyms, taxlist, missing-method (accepted_name), 3
synonyms, taxlist, numeric-method (accepted_name), 3
tax2traits, 28
tax2traits, taxlist-method (tax2traits), 28
taxize::tnrs(), 38
taxlist, 3–6, 8, 10–12, 14–23, 25–28, 30–39
taxlist-class, 29
taxlist-package, 3, 30
taxlist2taxmap, 30
taxlist2taxmap, taxlist-method (taxlist2taxmap), 30
taxmap, 30
taxmap2taxlist (taxlist2taxmap), 30
taxon_names, 31
taxon_names, taxlist-method (taxon_names), 31
taxon_names<-(taxon_names), 31
taxon_names<-, taxlist, data.frame-method (taxon_names), 31
taxon_relations, 33
taxon_relations, taxlist-method (taxon_relations), 33
taxon_relations<-(taxon_relations), 33
taxon_relations<-.,taxlist,data.frame-method
  (taxon_relations), 33
  taxon_traits, 35
  taxon_traits,taxlist-method
    (taxon_traits), 35
  taxon_traits<-.(taxon_traits), 35
  taxon_traits<-.,taxlist,data.frame-method
    (taxon_traits), 35
  taxon_views, 36
  taxon_views(), 34, 39
  taxon_views,taxlist-method
    (taxon_views), 36
  taxon_views<-.(taxon_views), 36
  taxon_views<-.,taxlist,data.frame-method
    (taxon_views), 36
  tnrs, 38
  tnrs,character-method(tnrs), 38
  tnrs,taxlist-method(tnrs), 38
  tv.home(), 39
  tv2taxlist, 39

  update_concept(taxon_relations), 33
  update_concept,taxlist,numeric-method
    (taxon_relations), 33
  update_name(taxon_names), 31
  update_name,taxlist,numeric-method
    (taxon_names), 31
  update_trait(taxon_traits), 35
  update_trait,taxlist,numeric-method
    (taxon_traits), 35

  vegdata-package, 39