Package ‘SoilTaxonomy’

November 16, 2023

Title A System of Soil Classification for Making and Interpreting Soil Surveys

Description Taxonomic dictionaries, formative element lists, and functions related to the maintenance, development and application of U.S. Soil Taxonomy. Data and functionality are based on official U.S. Department of Agriculture sources including the latest edition of the Keys to Soil Taxonomy. Descriptions and metadata are obtained from the National Soil Information System or Soil Survey Geographic databases. Other sources are referenced in the data documentation. Provides tools for understanding and interacting with concepts in the U.S. Soil Taxonomic System. Most of the current utilities are for working with taxonomic concepts at the "higher" taxonomic levels: Order, Suborder, Great Group, and Subgroup.

Version 0.2.4

Maintainer Andrew Brown <andrew.g.brown@usda.gov>
Depends R (>= 3.5)
License GPL (>= 3)
Encoding UTF-8
LazyLoad yes
Repository CRAN

URL https://github.com/ncss-tech/SoilTaxonomy,
    https://ncss-tech.github.io/SoilTaxonomy/

BugReports https://github.com/ncss-tech/SoilTaxonomy/issues
Imports stats, utils, stringr, data.table
Suggests testthat, knitr, markdown, markdown, soilDB, ape, data.tree
RoxygenNote 7.2.3
VignetteBuilder knitr
LazyData false

Date/Publication 2023-11-16 18:54:22 UTC
R topics documented:

- `code_to_level` Determine taxonomic level of a taxonomic letter code

**Description**

Determine taxonomic level of a taxonomic letter code

**Usage**

```r
code_to_level(code)
```

**Arguments**

- `code` A character vector of taxon codes (case sensitive)
**decompose_taxon_code**

Value

A character vector containing "order", "suborder", "greatgroup" or "subgroup"

Examples

```r
# order level code (1 character)
code_to_level("B")

# subgroup level code (4 characters)
code_to_level("ABCD")

# subgroup level code (5 characters, 4 uppercase + 1 lowercase)
code_to_level("IFFZh")
```

**decompose_taxon_code**  Decompose taxon letter codes

Description

Find all codes that logically comprise the specified codes. For instance, code "ABC" ("Anhyturbels") returns "A" ("Gelisols"), "AB" ("Turbels"), "ABC" ("Anhyturbels"). Use in conjunction with a lookup table that maps Order, Suborder, Great Group and Subgroup taxa to their codes (see `taxon_code_to_taxon` and `taxon_to_taxon_code`).

Usage

decompose_taxon_code(codes)

Arguments

codes A character vector of taxon codes to "decompose" – case sensitive

Details

Accounts for Keys that run out of capital letters (more than 26 subgroups) and use lowercase letters for a unique subdivision within the "fourth character position."

Value

A list with equal length to input vector; one character vector per element

See Also

`preceding_taxon_codes`, `taxon_code_to_taxon`, `taxon_to_taxon_code`
explainST

Explain a taxon name using formative elements

Description

Explain a taxon name using formative elements

Usage

explainST(x, format = c("text", "html"), viewer = TRUE)

Arguments

x a Subgroup, Great Group, Suborder or Order-level taxonomic name; matching is exact and case-insensitive

format output format: 'text' | 'html'

viewer show format = 'html' output in browser? default: TRUE

Value

a block of text, suitable for display in fixed-width font

Examples

cat(explainST("ids"), "\n\n") # -ids (order suffix)
cat(explainST("aridisols"), "\n\n") # Aridisols (order name)
cat(explainST("argids"), "\n\n") # Arg- (suborder)
cat(explainST("haplargids"), "\n\n") # Hap- (great group)
cat(explainST("typic haplargids"), "\n\n") # Typic (subgroup)
### extractSMR

*Extract Soil Moisture Regime from Subgroup or Higher Level Taxon*

**Description**

Extract Soil Moisture Regime from Subgroup or Higher Level Taxon

**Usage**

```
extractSMR(taxon, as.is = FALSE, droplevels = FALSE, ordered = TRUE)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>taxon</td>
<td>character. Vector of taxon names.</td>
</tr>
<tr>
<td>as.is</td>
<td>Return character labels rather than an (ordered) factor? Default: FALSE</td>
</tr>
<tr>
<td>droplevels</td>
<td>Drop unused levels? Default: FALSE</td>
</tr>
<tr>
<td>ordered</td>
<td>Create an ordinal factor? Default: TRUE</td>
</tr>
</tbody>
</table>

**Value**

an (ordered) factor of Soil Moisture Regimes, or character vector when as.is=TRUE

**Examples**

```
eextractSMR(c("aquic haploxeralfs", "typic epiaqualfs", "humic inceptic eutroperox"))
```

### FormativeElements

*Identify formative elements in taxon names at Soil Order, Suborder, Great Group or Subgroup level*

**Description**

Identify formative elements in taxon names at Soil Order, Suborder, Great Group or Subgroup level

**Usage**

```
FormativeElements(x, level = c("order", "suborder", "greatgroup", "subgroup"))
```

```
OrderFormativeElements(x)
SubOrderFormativeElements(x)
GreatGroupFormativeElements(x)
SubGroupFormativeElements(x)
```
get_ST_formative_elements(
  level = c("order", "suborder", "greatgroup", "subgroup")
)

Arguments

x  A character vector containing subgroup-level taxonomic names
level  one of c("order", "suborder", "greatgroup", "subgroup")

Value

A list containing $defs: a data.frame containing taxonomic elements, derivations, connotations and links. And $char.index: a numeric denoting the position where the formative element occurs in the search text x

get_ST_formative_elements(): a data.frame containing descriptors of formative elements used at the specified level

Author(s)

D.E. Beaudette, A.G. Brown

Examples

FormativeElements("acrudoxic plinthic kandiudults", level = "subgroup")
SubGroupFormativeElements("acrudoxic plinthic kandiudults")

FormativeElements("acrudoxic plinthic kandiudults", level = "greatgroup")
GreatGroupFormativeElements("acrudoxic plinthic kandiudults")

FormativeElements("acrudoxic plinthic kandiudults", level = "suborder")
SubOrderFormativeElements("acrudoxic plinthic kandiudults")

FormativeElements("acrudoxic plinthic kandiudults", level = "order")
OrderFormativeElements("acrudoxic plinthic kandiudults")

getChildTaxa  Get the lower (child) taxa for a taxon name or code

Description

Get the lower (child) taxa for a taxon name or code
**getFirstChildTaxon**

Usage

```r
getLastChildTaxon(level = c("order", "suborder", "greatgroup", "subgroup"))
```

Arguments

- `level`: Get child taxa from keys at specified level. One of: "order", "suborder", "greatgroup"

Value

A named list, where names are taxon codes and values are character vectors representing parent taxa

Examples

```r
# suborder children of "Mollisols"
getLastChildTaxon("Mollisols", level = "suborder")

# get all children within a great group, given a subgroup
getLastChildTaxon(getTaxonAtLevel("Ultic Haploxeralfs", "greatgroup"))
```
getParentTaxa

Value

A data.frame containing key (parent key), taxon (last taxon name), code (letter code), position (relative taxon position)

Examples

# get last taxa in suborder-level keys
x <- getLastChildTaxon(level = "suborder")

# proportion of keys where last taxon has "Hap" formative element
prop.table(table(grepl("^Hap", x$taxon)))

getParentTaxa

Get the higher (parent) taxa for a taxon name or code

Description

Must specify either taxon or code. taxon is used if both are specified.

Usage

g getParentTaxa(
  taxon = NULL,
  code = NULL,
  convert = TRUE,
  level = c("order", "suborder", "greatgroup", "subgroup")
)

Arguments

taxon A character vector of taxa (case-insensitive)

code A character vector of taxon codes (case sensitive)

convert Convert results from taxon codes to taxon names? Default: TRUE

level level Filter results to specific level? Default: "order","suborder","greatgroup","subgroup"

Value

A named list, where names are taxon codes and values are character vectors representing parent taxa
getTaxonAtLevel

Examples

getParentTaxa("ultic haploxeralfs")

getParentTaxa(code = c("ABCD", "DABC"))

getParentTaxa("folists", convert = FALSE)

g <<- getTaxonAtLevel

Description

Get the taxon name at the Soil Order, Suborder, Great Group or Subgroup level

Usage

getTaxonAtLevel(x, level = "order", simplify = TRUE)

Arguments

x A character vector containing subgroup-level taxonomic names
level one of c("order", "suborder", "greatgroup", "subgroup")
simplify Return a vector when level has length 1? Default: TRUE. Otherwise, a data.frame is returned.

Value

A named character vector of taxa at specified level, where names are the internal Soil Taxonomy letter codes. When length(level) > 1? a data.frame is returned with column names for each level.

Examples

# default gets the soil order
g <<- getTaxonAtLevel(c("typic haplargids", "typic glacistels")) #, level = "order")

# specify alternate levels
g <<- getTaxonAtLevel("humic haploxerands", level = "greatgroup")

# can't get subgroup (child) from great group (parent)
g <<- getTaxonAtLevel("udifolists", level = "subgroup")

# but can do parents of children
g <<- getTaxonAtLevel("udifolists", level = "suborder")
# specify multiple levels (returns a list element for each level)
getTaxonAtLevel("hapludolls", c("order", "suborder", "greatgroup", "subgroup"))

get_ST_family_classes  
*Get soil family / series differentiae and class names*

**Description**

All parameters to this function are optional (default NULL). If specified, they are used as filters.

**Usage**

```r
get_ST_family_classes(
  classname = NULL,
  group = NULL,
  name = NULL,
  chapter = NULL,
  page = NULL,
  multiline_sep = "\n",
  multiline_col = "criteria"
)
```

**Arguments**

- `classname` optional filtering vector; levels of ChoiceName column from NASIS metadata
- `group` optional filtering vector; one or more of: "Mineral Family", "Organic Family", "Mineral or Organic"
- `chapter` optional filtering vector for chapter number
- `page` optional filtering vector; page number (12th Edition Keys to Soil Taxonomy)
- `multiline_sep` default "\n" returns multiline_col column as a character vector concatenated with "\n". Use NULL for list
- `multiline_col` character vector of "multi-line" column names to concatenate. Default: "criteria"; use NULL for no concatenation.

**Details**

This is a wrapper method around the package data set ST_family_classes.
get_ST_features

Value

A `data.frame`

A subset of `ST_family_classes data.frame`

See Also

`ST_family_classes get_ST_features()`

Examples

```r
# get classes in chapter 17
str(get_ST_family_classes(chapter = 17))

# get classes on page 323
get_ST_family_classes(page = 323)

# get the description for the mesic temperature class from list column
str(get_ST_family_classes(classname = "mesic")$description)
```

get_ST_features

Get soil diagnostic horizons, characteristics and features

Description

All parameters to this function are optional (default NULL). If specified, they are used as filters.

Usage

```r
get_ST_features(
  group = NULL,
  chapter = NULL,
  name = NULL,
  page = NULL,
  multiline_sep = "\n",
  multiline_col = "criteria"
)
```

Arguments

- **group**: optional filtering vector; one of: "Surface", "Subsurface", "Mineral", "Organic", "Mineral or Organic"
- **chapter**: optional filtering vector for chapter number
- **name**: optional filtering vector; these are the "names" of features used in headers
- **page**: optional filtering vector; page number (12th Edition Keys to Soil Taxonomy)
isValidST

multiline_sep

default \"\n\" returns multiline_col column as a character vector concatenated with \"\n\". Use NULL for list.

multiline_col

character. vector of "multi-line" column names to concatenate. Default: "criteria"; use NULL for no concatenation.

Details

This is a wrapper method around the package data set ST_features.

Value

a subset of ST_features data.frame

See Also

ST_features ST_family_classes get_ST_family_classes()

Examples

# get all features
str(get_ST_features())

# get features in chapter 3
str(get_ST_features(chapter = 3))

# get features on pages 18, 19, 20
get_ST_features(page = 18:20)

# get the required characteristics for the mollic epipedon from list column
str(get_ST_features(name = "Mollic Epipedon")$criteria)

isValidST

Check for valid taxonomic level (Order, Suborder, Great Group, Subgroup)

Description

Checks needle for matches against a single level of Soil Taxonomy hierarchy: order, suborder, greatgroup, subgroup. Matches are case-insensitive.

Usage

isValidST(needle, level = c("order", "suborder", "greatgroup", "subgroup"))
Arguments

- needle: vector of taxa
- level: single level of Soil Taxonomy hierarchy; one of: "order", "suborder", "greatgroup", "subgroup"

Value

logical vector, same length as needle

Examples

isValidST('typic haploxeralfs', level = 'subgroup')

level_hierarchy

Order of Hierarchical Levels in Soil Taxonomy

Description

Creates an ordered factor such that different levels (the values used in level arguments to various SoilTaxonomy package functions) in the Soil Taxonomy hierarchy can be distinguished or compared to one another.

Usage

level_hierarchy(
  x = c("order", "suborder", "greatgroup", "subgroup", "family"),
  family = TRUE,
  as.is = FALSE
)

Arguments

- x: Passed as input to factor(); defaults to full set: "order", "suborder", "greatgroup", "subgroup", "family".
- family: Allow "family" as input in x? Used for validating inputs that must be a "taxon above family".
- as.is: Return x "as is" (after validation)? Shorthand for unclass(taxon_hierarchy()).

Details

The levels of Soil Taxonomy hierarchy include: "family", "subgroup", "greatgroup", "suborder", "order". The "order" is a level above "suborder". "subgroup" and above are "taxa above family". Note: "family" is always included as the "lowest" level when the result is an ordered factor, even when family-level input is disallowed by family=FALSE.
Value
An ordered factor with the values "order", "suborder", "greatgroup", "subgroup", or character when `as.is=TRUE`.

Examples

```r
# is great group a taxon above family?
level_hierarchy("greatgroup") > "family"

# is order lower level than suborder?
level_hierarchy("order") < "suborder"

# what levels are above or equal to a particular taxon's level?
level_hierarchy(as.is = TRUE)[level_hierarchy() >= taxon_to_level("aquisalids")]

## this produces an error (used for checking for taxa above family)
# level_hierarchy("family", family = FALSE)
```

---

**level_to_taxon**

*Get all taxa at specified level*

Description
Convenience method for getting taxa from `ST_unique_list`

Usage

```r
level_to_taxon(level = c("order", "suborder", "greatgroup", "subgroup"))
```

Arguments

- `level` character. One or more of "order", "suborder", "greatgroup", "subgroup"

Value
A character vector of taxa at the specified level

Examples

```r
# get all order and suborder level taxa
level_to_taxon(level = c("order","suborder"))
```
**newick_string**

**Generate Newick Tree Format Parenthetic Strings**

**Description**

This function generates Newick tree format strings for a single tree. Taxa are assigned relative positions within their parent to indicate the order that they "key out."

**Usage**

```r
newick_string(
  x = NULL,
  level = c("suborder", "greatgroup", "subgroup"),
  what = c("taxon", "code")
)
```

**Arguments**

- **x**: Optional: a taxon name to get children of.
- **level**: Level to build the tree at. One of "suborder", "greatgroup", "subgroup". Defaults to "suborder" when x is not specified. When x is specified but level is not specified, level is calculated from taxon_to_level(x).
- **what**: Either "taxon" (default; for taxon names (quoted for subgroups)) or "code"

**Details**

The output from this function is a character string with parenthetical format encoding a single tree suitable for input into functions such as `ape::read.tree()`. Multiple trees can be combined together in the file or text string supplied to your tree-parsing function of choice.

**Value**

character. A single tree in parenthetical Newick or New Hampshire format.

**Examples**

```r
if (requireNamespace("ape")) {
  par(mar = c(0, 0, 0, 0))

  # "fan"
  mytr <- ape::read.tree(text = newick_string(level = "suborder"))
  plot(mytr, "f", rotate.tree = 180, cex = 0.75)

  # "cladogram"
  mytr <- ape::read.tree(text = newick_string("durixeralfs", level = "subgroup"))
  plot(mytr, "c")

  # "cladogram" (using taxon codes instead of subgroups)
}
```
mytr <- ape::read.tree(text = newick_string("xeralfs", level = "subgroup", what = "code"))
plot(mytr, "c")

dev.off()
}

---

**parent_level**

**Parent/Child Hierarchy**

**Description**

Parent/Child Hierarchy

**Usage**

```r
parent_level(level, n = 1)
child_level(level, n = 1)
```

**Arguments**

- **level** character. Initial level name of a taxon. Vectors include values that are one of: "order", "suborder", "greatgroup", "subgroup", "family"
- **n** Number of levels above/below (parent/child). Default: 1

**Value**

character. Level name of parent or child at specified level above input level.

**Examples**

```r
parent_level('subgroup')
child_level('greatgroup')
parent_level('family', 3)
# no level above order
parent_level('family', 5)
```
parse_family

Parse components of a "family-level" taxon name

Description

Parse components of a "family-level" taxon name

Usage

parse_family(family, column_metadata = TRUE, flat = TRUE)

Arguments

family character. vector of taxonomic families, e.g. "fine-loamy, mixed, semiactive, mesic ultic haploxeralfs"

column_metadata logical. include parsed NASIS physical column names and values from family taxon components? Default: TRUE requires soilDB package.

flat logical Default: TRUE to return concatenated family-level classes for "taxminalogy" and "taxfamother"? Alternately, if FALSE, list columns are returned.

Value

a data.frame containing column names: "family" (input), "subgroup" (parsed taxonomic subgroup), "subgroup_code" (letter code for subgroup), "class_string" (comma-separated family classes), "classes_split" (split class_string vector stored as list column).

In addition, the following column names are identified and returned based on NASIS (National Soil Information System) metadata (via soilDB package):

- "taxpartsize","taxpartsizemod","taxminalogy","taxceactcl","taxreaction","taxtempcl","taxfamhahatmatcl","taxfamother","taxsubgrp","taxgreatgroup","taxsuborder","taxorder"

Examples

if (requireNamespace('soilDB')) {
  families <- c("Fine, kaolinitic, thermic typic kanhapludults",
                "fine-loamy, mixed, semiactive, mesic ultic haploxeralfs",
                "euic, thermic typic haplosapris",
                "coarse-loamy, mixed, active, mesic aquic dystrudepts")

  # inspect parsed list result
  str(parse_family(families))
}

}
preceding_taxon_codes  Get taxon codes of preceding taxa

Description

Find all codes that logically precede the specified codes. For instance, code "ABC" ("Anhyturbels") returns "AA" ("Histels") "ABA" ("Histoturbels") and "ABB" ("Aquiturbels"). Use in conjunction with a lookup table that maps Order, Suborder, Great Group and Subgroup taxa to their codes (see taxon_code_to_taxon and taxon_to_taxon_code).

Usage

preceding_taxon_codes(codes)

Arguments

codes  A character vector of codes to calculate preceding codes for

Details

Accounts for Keys that run out of capital letters (more than 26 subgroups) and use lowercase letters for a unique subdivision within the "fourth character position."

Value

A list with equal length to input vector; one character vector per element

See Also

decompose_taxon_code, taxon_code_to_taxon, taxon_to_taxon_code

Examples

preceding_taxon_codes(c("ABCDe", "BCDEf"))
**relative_taxon_code_position**

*Determine relative position of taxon within Keys to Soil Taxonomy (Order to Subgroup)*

---

**Description**

The relative position of a taxon is 
[number of preceding Key steps] + 1, or NA if it does not exist in the lookup table.

**Usage**

`relative_taxon_code_position(code)`

**Arguments**

- `code` 
  A character vector of taxon codes to determine the relative position of.

**Value**

A numeric vector with the relative position of each code with respect to their individual Keys.

**Examples**

```r
# "ABCD" -> "Gypsic Anhyturbels", relative position 7
# "WXYZa" does not exist, theoretical position is 97
# "BAD" -> "Udifolists", relative position is 5

relative_taxon_code_position(c("ABCD", "WXYZa", "BAD"))
# [1] 7 NA 5
```

---

**SoilTaxonomyLevels**

*Get (Ordered) Factors based on Soil Taxonomy Key position*

---

**Description**

Get (Ordered) Factors based on Soil Taxonomy Key position
Usage

SoilTaxonomyLevels(
  level = c("order", "suborder", "greatgroup", "subgroup"),
  as.is = FALSE,
  ordered = TRUE
)

SoilMoistureRegimeLevels(as.is = FALSE, ordered = TRUE)

SoilTemperatureRegimeLevels(as.is = FALSE, ordered = TRUE)

Arguments

  level One of: "order", "suborder", "greatgroup", "subgroup"
  as.is Return character labels rather than an (ordered) factor? Default: FALSE
  ordered Create an ordinal factor? Default: TRUE

Value

  an (ordered) factor or character vector (when as.is=TRUE)

Examples

SoilTaxonomyLevels("order")

SoilTaxonomyLevels("order", ordered = FALSE)

SoilTaxonomyLevels("order", as.is = TRUE)

SoilTaxonomyLevels("suborder")

ST  

Soil Taxonomy Hierarchy

Description

  The first 4 levels of the US Soil Taxonomy hierarchy (soil order, suborder, greatgroup, subgroup),
  presented as a data.frame (denormalized) and a list of unique taxa.

Usage

  data(ST)

Format

  An object of class data.frame with 2665 rows and 9 columns.
Details

Ordered based on the unique letter codes denoting taxa from the 13th edition of the Keys to Soil Taxonomy.

References


<table>
<thead>
<tr>
<th>ST_family_classes</th>
<th>Family-level Classes for Soil Taxonomy</th>
</tr>
</thead>
</table>

Description

A database of family-level class names for Soil Taxonomy.

Usage

data(ST_family_classes)

Format

An object of class data.frame with 193 rows and 8 columns.

References

ST_features  
*Epipedons, Diagnostic Horizons, Characteristics and Features in Soil Taxonomy*

**Description**


**Usage**

```r
data(ST_features)
```

**Format**

An object of class `data.frame` with 84 rows and 6 columns.

**References**


---

ST_formative_elements  
*Formative Elements used by Soil Taxonomy*

**Description**

A database of formative elements used by the first 4 levels of US Soil Taxonomy hierarchy (soil order, suborder, greatgroup, subgroup).

**Usage**

```r
data(ST_formative_elements)
```

**Format**

An object of class `list` of length 4.

**References**

ST_higher_taxa_codes_12th

Letter Code Lookup Table for Position of Taxa within the Keys to Soil Taxonomy (12th Edition)

Description

A lookup table mapping unique taxonomic Order, Suborder, Great Group and Subgroups to letter codes that denote their logical position within the Keys.

Usage

data(ST_higher_taxa_codes_12th)

Format

An object of class data.frame with 3082 rows and 2 columns.

Details

The lookup table has been corrected to reflect errata that were posted after the print publication of the 12th Edition Keys, as well as typos in the Spanish language edition.

References


ST_higher_taxa_codes_13th

Letter Code Lookup Table for Position of Taxa within the Keys to Soil Taxonomy (13th Edition)

Description

A lookup table mapping unique taxonomic Order, Suborder, Great Group and Subgroups to letter codes that denote their logical position within the Keys.

Usage

data(ST_higher_taxa_codes_13th)
taxonTree

Format

An object of class data.frame with 3153 rows and 2 columns.

References


taxonTree

Create a data.tree Object from Taxon Names

Description

This function takes one or more taxon names and taxonomic levels as input.

Usage

taxonTree(
  taxon,
  level = c("order", "suborder", "greatgroup", "subgroup"),
  root = "Soil Taxonomy",
  verbose = TRUE,
  special.chars = c("|--", "|", "|", "-"),
  file = "",
  ...
)

Arguments

taxon A vector of taxon names

level One or more of: "order", "suborder", "greatgroup", "subgroup". The lowest level is passed to getChildLevel() to generate the leaf nodes.


verbose Print tree output? Default: TRUE

special.chars Characters used to print the tree to console. Default: c("|--", "|", "|", "-"). For fancy markup try: c("\u251c", "\u2502", "\u2514", "\u2500 ")

file Optional: path to output file. Default: "" prints to standard output connection (unless redirected by sink())

... Additional arguments to data.tree::as.Node.data.frame()

Details

A subclass of data.tree Node object is returned. This object has a custom print() method
Value

A SoilTaxonNode (subclass of data.tree Node) object (invisibly). A text representation of the tree is printed to stdout when verbose=TRUE.

Examples

    # hapludults and hapludalfs (to subgroup level)
    taxonTree(c("hapludults", "hapludalfs"))

    # alfisols suborders and great groups
    taxonTree("alfisols", root = "Alfisols", level = c("suborder", "greatgroup"))

```
taxon_code_to_taxon  Convert taxon code to taxon name

Description

Convert taxon code to taxon name

Usage

taxon_code_to_taxon(code)

Arguments

code  A character vector of Taxon Codes

Value

A character vector of matching Taxon Names

See Also

decompose_taxon_code, preceding_taxon_codes, taxon_to_taxon_code

Examples

    taxon_code_to_taxon(c("ABC", "XYZ", "DAB", NA))
```
### taxon_to_level

**Determine taxonomic level of specified taxa**

**Description**

Taxa that resolve to a subgroup level taxon and contain a comma "," are assumed to be "family"-level.

**Usage**

```r
taxon_to_level(taxon)
```

**Arguments**

- `taxon` character vector of taxon names at Order, Suborder, Great Group or Subgroup level.

**Value**

character of taxonomic hierarchy levels (such as "order", "suborder", "greatgroup", "subgroup", "family") for each element of input vector.

**Examples**

```r
# get the taxonomic levels for various taxa
taxon_to_level(c("gelisols", NA, "foo", "typic folistels", "folistels"))
```

### taxon_to_taxon_code

**Convert taxon name to taxon code**

**Description**

Convert taxon name to taxon code

**Usage**

```r
taxon_to_taxon_code(taxon)
```

**Arguments**

- `taxon` A character vector of taxon names, case insensitive
Value
A character vector of matching taxon codes

See Also
decompose_taxon_code, preceding_taxon_codes, taxon_code_to_taxon

Examples

taxon_to_taxon_code(c("Anhyturbels", "Foo", "Cryaquands", NA))
Index

* datasets
  ST, 20
  ST_family_classes, 21
  ST_features, 22
  ST_formative_elements, 22
  ST_higher_taxa_codes_12th, 23
  ST_higher_taxa_codes_13th, 23

child_level (parent_level), 16
code_to_level, 2
decompose_taxon_code, 3, 18, 25, 27
explainST, 4
extractSMR, 5

FormativeElements, 5
get_ST_family_classes, 10
get_ST_features, 11
get_ST_formative_elements
  (FormativeElements), 5
getChildTaxa, 6
getLastChildTaxon, 7
getParentTaxa, 8
getTaxonAtLevel, 9
GreatGroupFormativeElements
  (FormativeElements), 5
isValidST, 12

level_hierarchy, 13
level_to_taxon, 14

newick_string, 15

OrderFormativeElements
  (FormativeElements), 5

parent_level, 16
parse_family, 17

preceding_taxon_codes, 3, 18, 25, 27
relative_taxon_code_position, 19
SoilMoistureRegimeLevels
  (SoilTaxonomyLevels), 19
SoilTaxonomyLevels, 19
SoilTemperatureRegimeLevels
  (SoilTaxonomyLevels), 19
ST, 20
ST_family_classes, 21
ST_features, 22
ST_formative_elements, 22
ST_higher_taxa_codes_12th, 23
ST_higher_taxa_codes_13th, 23
ST_unique_list (ST), 20
SubGroupFormativeElements
  (FormativeElements), 5
SubOrderFormativeElements
  (FormativeElements), 5
taxon_code_to_taxon, 3, 18, 25, 27
taxon_to_level, 26
taxon_to_taxon_code, 3, 18, 25, 26
taxonTree, 24