Package: n2khabmon (via r-universe)

August 13, 2024

Title Prepare and Manage N2KHAB Monitoring Schemes

Version 0.3.0

Description n2khabmon is an R package with utilities to prepare and manage Flemish monitoring schemes regarding Natura 2000 habitats and regionally important biotopes (RIBs).

License GPL (>= 3)

URL https://inbo.github.io/n2khabmon, https://github.com/inbo/n2khabmon

BugReports https://github.com/inbo/n2khabmon/issues

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.1

Imports assertthat, curl, dplyr, git2rdata, n2khab (>= 0.10.0), remotes, rlang, stringr, tidyr

Remotes inbo/n2khab

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

RemoteUrl https://github.com/inbo/n2khabmon

RemoteRef HEAD

RemoteSha 57820de61a57ed68f315bd86cafb03387b72ec5e

Contents

`namelist` *Documentation of included data source 'namelist'*

Description

'namelist' is a data source in the vc-format which provides names and (optionally) shortnames for IDs/codes used in other data sources. Multiple languages are supported.

Format

A vc-formatted data source. As such, it corresponds to a data frame with many rows and 4 variables:

code A code used elsewhere.

lang An IETF BCP 47 language tag, such as `"en"` or `"nl"`, to identify the language of `name` and `shortname`.

name The name corresponding to `code` and `lang`.

shortname Optionally, a shorter variant of `name`.

Typical way of loading

```
read_namelistmon()
read_namelistmon(lang = "nl")
```

Corresponding datafiles in the installed package

`textdata/namelist.tsv`

`textdata/namelist.yml`

Source

The 'namelist' data source has got its contents from the sources of several referencelists in package `n2khab` (see the source of those).

See Also

`read_namelistmon`

Other n2khabmon-referencelists: `scheme_types`, `schemes`

`read_namelistmon` *Return the 'namelist' data source as a tibble*

Description

Returns the included data source `namelist` as a `tibble`, by default filtered according to English names and shortnames.

Usage

```
read_namelistmon(
  path = pkgdatasource_path("textdata/namelist", ".yml"),
  file = "namelist",
  lang = "en"
)
```

Arguments

`path`	Location of the data source. The default is to use the location of the data source as delivered by the installed package.
`file`	The filename of the data source, without extension. The default is to use the file delivered by the installed package.
`lang`	An IETF BCP 47 language tag, such as `"en"` or `"nl"`, to specify the language of `name` and `shortname` to be returned in the tibble. If `lang = "all"`, the full `namelist` tibble is returned, i.e. containing all languages.

Details

`namelist` is a data source in the vc-format which provides names and (optionally) shortnames for IDs/codes used in other data sources.

`read_namelistmon()` reads it and returns it as a `tibble`. A tibble is a data frame that makes working in the tidyverse a little easier. By default, the data version delivered with the package is used and only English names (`lang = "en"`) are returned.

Value

The `namelist` data frame as a `tibble`, filtered according to the `lang` argument. See `namelist` for documentation of the tibble's contents.

Recommended usage

```
read_namelistmon()
```

```
read_namelistmon(lang = "nl")
```

See Also

namelist n2khab::read_namelist

Other reading functions for n2khabmon-referencelists: read_scheme_types(), read_schemes()

Examples

```
read_namelistmon()
read_namelistmon(lang = "nl")
```

---

read_schemes *Return the 'schemes' data source as a tibble with names & shortnames*

Description

Returns the included data source schemes as a tibble. Names and shortnames from namelist are added, in English by default.

Usage

```
read_schemes(
  path = pkgdatasource_path("textdata/schemes", ".yml"),
  file = "schemes",
  file_namelist = "namelist",
  lang = "en"
)
```

Arguments

path	Location of the data sources schemes and namelist. The default is to use the location of the data sources as delivered by the installed package.
file	The filename of the schemes data source, without extension. The default is to use the file delivered by the installed package.
file_namelist	The filename of the namelist data source, without extension. The default is to use the file delivered by the installed package.
lang	An IETF BCP 47 language tag, such as "en" or "nl", to specify the language of names & shortnames to be returned in the tibble.

Details

schemes is a data source in the vc-format which provides a list of (monitoring) schemes for N2KHAB monitoring programmes or other N2KHAB projects, together with defining attributes and optional information. A 'scheme' refers to a monitoring or research setup that determines which types (habitat/RIBs) are to be investigated for a question or for a bunch of related questions.

read_schemes() reads the schemes data source, adds names + shortnames and returns it as a tibble. A tibble is a data frame that makes working in the tidyverse a little easier. By default, the data version delivered with the package is used and English names (lang = "en") are returned for scheme, programme, attributes and tags.

Value

The schemes data frame as a tibble, with names & shortnames added for scheme, programme, attributes and tags according to the lang argument. The tibble has 25 variables. See schemes for documentation of the data-source's contents. See namelist for the link between codes or other identifiers and the corresponding names (and shortnames).

The added names and shortnames are represented by the following variables:

- scheme_name
- scheme_shortname
- programme_name
- attribute_1_name
- attribute_1_shortname
- attribute_2_name
- attribute_2_shortname
- attribute_3_name
- attribute_3_shortname
- tag_1_name
- tag_1_shortname
- tag_2_name
- tag_2_shortname
- tag_3_name
- tag_3_shortname

The added names and shortnames for scheme, programme and attributes are *factors* with their level order according to that of the scheme, programme or attribute variable.

Recommended usage

```
read_schemes()
read_schemes(lang = "nl")
```

Anticipating conflicts with n2khab (where this function is deprecated)

An efficient way with base R to avoid function masking and conflict warnings when attaching both {n2khab} and {n2khabmon}, regardless of the order in which they're loaded, is by specifying something as below in your script, at least *before* loading {n2khab}:

```
conflictRules("n2khab", exclude = c("read_schemes", "read_scheme_types"))
```

See Also

schemes

Other reading functions for n2khabmon-referencelists: read_namelistmon(), read_scheme_types()

Examples

```
read_schemes()
read_schemes(lang = "nl")
```

`read_scheme_types` *Return the 'scheme_types' data source as a tibble*

Description

Returns the included data source `scheme_types` as a `tibble`. Names and shortnames from `namelist` are optionally added, in English by default.

Usage

```
read_scheme_types(
  path = pkgdatasource_path("textdata/scheme_types", ".yml"),
  file = "scheme_types",
  file_namelist = "namelist",
  lang = "en",
  extended = FALSE
)
```

Arguments

`path`	Location of the data sources `scheme_types`, `schemes`, `types` and `namelist`. The default is to use the location of the data sources as delivered by the installed package.
`file`	The filename of the `scheme_types` data source, without extension. The default is to use the file delivered by the installed package.
`file_namelist`	The filename of the `namelist` data source, without extension. The default is to use the file delivered by the installed package.
`lang`	An IETF BCP 47 language tag, such as `"en"` or `"nl"`, to specify the language of names & shortnames to be returned in the tibble.
`extended`	Should names & shortnames be added for scheme, programme, scheme attributes, type, typeclass and tags of scheme and type?

Details

`scheme_types` is a data source in the vc-format which lists the types (using the type-code from `types`) that belong to each N2KHAB (monitoring or research) scheme (using the scheme-code from `schemes`). It also defines typegroup memberships of the types within specific schemes, if applicable.

`read_scheme_types()` reads the `scheme_types` data source, optionally adds names + shortnames (always done for the typegroup) and returns it as a `tibble`. A tibble is a data frame that makes working in the tidyverse a little easier. By default, the data version delivered with the package is used and English names (`lang = "en"`) are returned.

Value

The scheme_types data frame as a tibble, with names & shortnames added for the typegroup variable and optionally for scheme, programme, scheme attributes, type and attributes & tags of scheme and type, all according to the lang argument. The tibble has either 5 or many variables, depending on the extended argument. See scheme_types for documentation of the data-source's contents. See namelist for the link between codes or other identifiers and the corresponding names (and shortnames).

The *optionally* added names and shortnames are represented by the following variables:

- scheme_name
- scheme_shortname
- programme_name
- attribute_1_name
- attribute_1_shortname
- attribute_2_name
- attribute_2_shortname
- attribute_3_name
- attribute_3_shortname
- schemetag_1_name
- schemetag_1_shortname
- schemetag_2_name
- schemetag_2_shortname
- schemetag_3_name
- schemetag_3_shortname
- type_name
- type_shortname
- typeclass_name
- hydr_class_name
- hydr_class_shortname
- groundw_dep_name
- groundw_dep_shortname
- flood_dep_name
- flood_dep_shortname
- typetag_1_name
- typetag_1_shortname
- typetag_2_name
- typetag_2_shortname
- typetag_3_name
- typetag_3_shortname

The added names and shortnames for scheme, programme, attributes and typegroup are *factors* with their level order according to that of the scheme, programme, attribute or typegroup variable.

Recommended usage

```
read_scheme_types()
read_scheme_types(lang = "nl")
```

Anticipating conflicts with n2khab (where this function is deprecated)

An efficient way with base R to avoid function masking and conflict warnings when attaching both {n2khab} and {n2khabmon}, regardless of the order in which they're loaded, is by specifying something as below in your script, at least *before* loading {n2khab}:

```
conflictRules("n2khab", exclude = c("read_schemes", "read_scheme_types"))
```

See Also

scheme_types

Other reading functions for n2khabmon-referencelists: read_namelistmon(), read_schemes()

Examples

```
read_scheme_types()
read_scheme_types(lang = "nl")
```

---

schemes *Documentation of included data source 'schemes'*

Description

'schemes' is a data source in the vc-format which provides a list of (monitoring) schemes for N2KHAB monitoring programmes, together with defining attributes and optional information. The codes of schemes, programmes, attributes and tags are explained in the data source namelist (which can accommodate multiple languages).

Format

A vc-formatted data source. As such, it corresponds to a data frame with 10 variables:

scheme Code of the scheme, as a factor. This is the ID for use in diverse workflows and datasets. Corresponding names and shortnames in multiple languages are to be found in namelist. Contains no duplicates!

A 'scheme' refers to a monitoring setup that determines which types (habitat/RIBs) are to be investigated.

programme Code of the programme to which the scheme belongs. The programme's code is explained by namelist. Is a factor.

A 'programme' refers to a N2KHAB monitoring programme. One programme can correspond to multiple schemes. At least the monitoring programmes MHQ and MNE are present.

attribute_1 The first defining attribute of the scheme. Typically, the code is explained by `namelist`. Is a factor.

- In MNE, this is used to declare the 'compartment superscheme' (scheme collection) to which the scheme belongs, and which is named after the environmental compartment (however note that the surfacewater superscheme comprises two environmental compartments).
- In MHQ, this is used to define the target habitat type of the monitoring scheme.

attribute_2 A second defining attribute of the scheme (if needed). Typically, the code is explained by `namelist`. Is a factor.

- In MNE, this provides the code of the environmental pressure to which the scheme is related. It must be represented in `env_pressures`.

attribute_3 A third defining attribute of the scheme (if needed). Typically, the code is explained by `namelist`. Is a factor.

- In MNE, this is used to declare an **optional** partitioning of the scheme that would be constituted by attributes 1 and 2 alone (environmental compartment and pressure), which will hence divide the concerned types in at least two groups. This typically has to do with differences in the considered environmental variables or compatibility of interpretation. It is *not* meant to distinguish between different types or typegroups for inferences, so types are kept in the same scheme as long as the measured values have compatibility of interpretation. To declare which types or typegroups should be distinguished in inferences, use the `typegroup` variable in the `scheme_types` data source!

spatial_restriction Optional further defining specification of a scheme (in English): spatial restrictions, superposed on the restrictions already generated by the attributes.

notes Optional. In English. For additional notes on the scheme definition, if necessary.

tag_1 Optional tag, e.g. a categorization ID explained by `namelist`.

- In MNE, this is used to tag which schemes are focal schemes and which ones are secondary schemes.
- In MHQ, this is used to separate aquatic and terrestial monitoring schemes.

tag_2 Optional tag, e.g. a categorization ID explained by `namelist`.

tag_3 Optional tag, e.g. a categorization ID explained by `namelist`.

Typical way of loading

```
read_schemes()
```

```
read_schemes(lang = "nl")
```

Corresponding datafiles in the installed package

`textdata/schemes.tsv`

`textdata/schemes.yml`

Source

- For the MNE-schemes: a vc-formatted, Dutch-language data source which can be found in the n2khab Github repository: `misc/generate_textdata/rawraw_data/10_compmeetnet_types_milieudrukken`. This data source was originally generated in the n2khab-mne-selections repository (‘`https://gitlab.com/florisvdh`
- More information on the MHQ-schemes can be found in Westra *et al.* (2022).

See Also

`read_schemes`

Other n2khabmon-referencelists: `namelist`, `scheme_types`

`scheme_types` *Documentation of included data source 'scheme_types'*

Description

'scheme_types' is a data source in the vc-format which lists the types (using the type-code from `types`) that belong to each N2KHAB monitoring scheme (using the scheme-code from `schemes`). It also defines typegroup memberships of the types within specific schemes, if applicable. The codes of schemes and types (and optionally: typegroups) are explained in the data source `namelist` (which can accommodate multiple languages).

Format

A vc-formatted data source. As such, it corresponds to a data frame with 3 variables:

scheme Code of the scheme, as a factor, and explained by `namelist`. It must be represented in `schemes`.

type Code of the type, as a factor, and explained by `namelist`. It must be represented in `types`.

typegroup An *optional* code (and optionally explained by `namelist`), declaring the typegroup to which a type belongs *within the specified scheme*. Typegroups point out which types are considered together as a group in inferences for the scheme at hand.

Each combination of `scheme` and `type` must be unique.

Typical way of loading

```
read_scheme_types()
read_scheme_types(lang = "nl")
```

Corresponding datafiles in the installed package

`textdata/scheme_types.tsv`

`textdata/scheme_types.yml`

Source

- For the MNE-schemes:
  - the link between schemes and types comes from a vc-formatted, Dutch-language data source which can be found in the n2khab Github repository: `misc/generate_textdata/rawraw_data/10_compme`
    This data source was originally generated in the n2khab-mne-selections repository (`https://gitlab.com/floris`
  - the declaration of typegroups comes from this googlesheet. Currently, the typegroups are kept up to date both in the googlesheet and in the data source.
- For the MHQ-schemes, the list of sampled types is based on the report of Westra *et al.* (2022).

See Also

`read_scheme_types`

Other n2khabmon-referencelists: `namelist`, `schemes`

Index