Package: annex (via r-universe)

May 19, 2026

Title IEA EBC Annex86 Data Analysis Package

Version 0.2-14

Date 2025-04-03

Description The main aim of the package is to streamline and standardize processing and storing data sets for the IEA EBC Annex 86 project.

Depends R (>= 4.0.0), utils

Imports Formula, zoo, tidyr, dplyr, openxlsx, crayon

Suggest readxl

License GPL-2 | GPL-3

URL https://github.com/IEA-EBC-Annex86/annex/

BugReports https://github.com/IEA-EBC-Annex86/annex/issues

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

NeedsCompilation no

Config/pak/sysreqs libicu-dev

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

Date/Publication 2025-05-15 18:43:15 UTC

RemoteUrl https://github.com/IEA-EBC-Annex86/annex

RemoteRef HEAD

RemoteSha 7acc0592e2089876a7462ccc5f09e90e35359774

Contents

`annex` *Annex Creator Function*

Description

Creates an object of class `c("annex", "data.frame")` required to calculate the statistics.

Usage

```
annex(formula, data, tz, duplicate.action = NULL, meta = NULL, verbose = FALSE)

## S3 method for class 'annex'
head(x, ...)

## S3 method for class 'annex'
tail(x, ...)
```

```
## S3 method for class 'annex'
subset(x, ...)

## S3 method for class 'annex_stats'
head(x, ...)

## S3 method for class 'annex_stats'
tail(x, ...)

## S3 method for class 'annex_stats'
subset(x, ...)
```

Arguments

formula	the formula to specify how the data set is set up. See 'Details' for more information.
data	data.frame containing the obervations/data.
tz	character, time zone definition (e.g., "Europe/Berlin" or "UTC"); required. OlsonNames() returns a list of possible time zones. The correct time zone is important to properly calculate month and time of day.
duplicate.action	NULL or a function which returns a single numeric value. Used to handle possible duplicates, see 'Details'.
meta	NULL (default) or a list with information about study, home, and room (see section 'Duplicates').
verbose	logical, defaults to FALSE. Can be set to TRUE to increase verbosity.
x	object of class annex.
...	arguments to be passed to or from other methods.

Details

In case the data set provided on data does only contain data of one study, home, and room, the fomula has two parts, looking e.g., as follows:

- T + RH ~ datetime

The left hand side of the formula (left of ~) specifies the names of the variables of the observations to be processed, the right hand side is the name of the variable containing the time information (must be of class POSIXt). In this case, the meta argument is required to provide information about the study, home, and room.

If the grouping information is already in the data set, the analysis can be performed depending on the group information, typically:

- T + RH ~ datetime | study + home + room

The latter allows to process observations from different studies, homes, and/or rooms all in one go.

Duplicates

Duplicated records can distort the statistics and should be handled properly. For each unique `study, home, room` only one observation (row) for a specific date and time should exist.

As there is no general way to deal with such duplicates, the function `annex` (as well as `annex_prepare`) by default throws a warning for the user if such duplicates exist (`duplicate.action = NULL`; default argument).

However, the package allows the user to provide a custom `duplicate.action` function, e.g., `mean, min, max, ...` This function must return one single numeric value (or an NA) when applied to a vector. If a function is provided, the `annex` function does the following:

- Checks if there are any duplicates. If not, no changes are made. Else ...
- Checking if the function is valid (returns single numeric or NA). If not, an error will be thrown.
- Takes the measurements of each duplicate; if all values are missing, an `NA` will be returned. Else the users `duplicate.action` is applied to all remaining non-missing values. I.e., if `duplicate.action = mean` the average of all non-missing values will be used.

Author(s)

Reto Stauffer

See Also

annex_prepare, annex_stats

Examples

```
# Create artificial data set for testing; typically this information is read
# from a file or any other data connection.
data <- data.frame(datetime = as.POSIXct("2022-01-01 00:00", tz = "Europe/Berlin") + -10:10 * 3600,
                   T  = round(rnorm(21, mean = 20, sd = 2), 2),
                   RH = round(runif(21, 40, 100), 2))

res1 <- annex(T + RH ~ datetime, data = data,
              meta = list(study = "example", home = "ex", room = "BED"),
              tz = "Europe/Berlin")
head(res1, n = 3)

# The meta information can also be added to `data` removing the need
# to specify the `meta` argument and allows to mix data from different
# studies and rooms. Appending study, room, and home to `data`:
data <- transform(data,
                  study = "example",
                  home  = "ex",
                  room  = "BED")
head(data)
res2 <- annex(T + RH ~ datetime | study + home + room,
              data = data, tz = "Europe/Berlin")
head(res2, n = 3)
```

annex_add_year_month_and_tod *Calculate year, month and time of day*

Description

Calculates year, month and the time of day categories based on input argument `x` with respect to the time zone specified by the user.

Usage

```
annex_add_year_month_and_tod(x, tz)
```

Arguments

`x`	object of class `POSIXt`.
`tz`	time zone (character). Important to properly calculate month and time of day.

Value

List with three elements `year` (integer), `month` (factor) and `tod` (factor).

Author(s)

Reto Stauffer

Examples

```
x <- as.POSIXct("2022-01-01", tz = "UTC") + 0:10 * 3600
annex:::annex_add_year_month_and_tod(x, tz = "Europe/Berlin")
annex:::annex_add_year_month_and_tod(x, tz = "US/Central")
```

annex_check_config *Checking Annex Config*

Description

An 'annex config object' is a simple data.frame which can be created using Rs standard features. The function is checking if the content of the object matches the requirements of being an annex config object. This check will be performed in `link{annex()}` automatically, but can also be done by the user manually.

Usage

```
annex_check_config(x)
```

Arguments

x	object of class data.frame to be checked.

Details

The function checks if the config object is set up properly and contains the required information for preparing the annex data.

Throws errors if:

- the input is not a data.frame
- variables are missing ('column', 'variable', 'unit', 'study', 'home', 'room')
- configuration for variable = "datetime" is missing
- there is no definition for variables (datetime only)
- the config contains missing values in the required variables
- column name for variable = "datetime" is missing
- the variable 'column' is not unique
- there are duplicated entries for the combination of variable/study/home/room (must be unique)
- variables study, home or room contain contain non-allowed characters. Must only contain letters (lowercase or uppercase), numbers, and underscores. Must start with a letter.

Value

Invisibly returns a (possibly modified) version of x containing only the required columns in a specific order.

Author(s)

Reto Stauffer

---

annex_check_stats_object

Checking Annex Stats to XML

Description

Used for checking/validating annex_stat objects; used internally by annex_write_stats() to ensure that what the users (try to) store to the final XLSX files is what is expected for the final output file.

Usage

```
annex_check_stats_object(x)
```

Arguments

x	object of class annex_stat as returned by annex_stats().

Details

The following will be checked:

- Input is of correct type and has at least one observation.
-
-
-
-

Value

No return, will throw an error if something does not match the expected file format.

Author(s)

Reto Stauffer

---

annex_countries *Country codes (ISO 3166) alpha-2 and alpha-3*

Description

Country codes (ISO 3166) alpha-2 and alpha-3

Usage

```
annex_countries()
```

Value

Returns the data set ISO3166 shipped with the package (see ?ISO3166 for details).

Author(s)

Reto Stauffer

annex_dist *Get (Mixed) Empirical Distribution*

Description

Get (Mixed) Empirical Distribution

Usage

```
annex_dist(x, ..., verbose = TRUE)
```

Arguments

x	a named vector or data.frame with empirical quantiles and sample size (optional). See Section 'Details' for more information.
verbose	Logical, if TRUE some additional output is shown.
'...'	see Section 'Details'.

Details

This function returns an annex_edist (annex empirical distribution) object which can be based on a single empirical distribution, or a (weighted) mixed distribution. annex_stats stores the empirical distribution of our measurements given a series of quantiles as well as the sample size the quantiles are based on. This function allows a series of different inputs.

Named numeric vector: x can be a named numeric vector. In this case, all the names of the vector must be unique and follow ^p[0-9]{1,3}(\.[0-9]{,2})?$. Some examples: p00 (minimum), p2 (2_th_ percentile or 0.02 quantile), p02.5 (2.5_th_ percentile or 0.025 quantile), p05 (5_th_ percentile or quantile 0.05) etc. up to p100 (maximum). Minimum (p00) and maximum (p100) *must* be given, the quantiles in between are handled flexible.

Author(s)

Reto Stauffer

annex_dist_process_input *Helper Function to Prepare User Input on* annex_dist

Description

Helper function handling the different input types allowed when calling annex_process_input. Check man page for annex_process_input for details.

Usage

```
annex_dist_process_input(x, ..., verbose = TRUE)
```

Value

Returns prepared (standardized) list for further processing.

Author(s)

Reto Stauffer

---

annex_handle_duplicates

Internal Function for Handling Possible Duplicates

Description

This function is called inside `annex()`.

Usage

```
annex_handle_duplicates(x, formula, duplicate.action, verbose = FALSE)
```

Arguments

`x`	data.frame to be processed.
`formula`	object of class `Formula`, the formula provided to `annex()`.
`duplicate.action`	can be `NULL` (no dedicated handling for duplicates) of a function. If function it must return a single numeric value, will be tested to be able to provide a useful error for the user if needed.
`verbose`	logical, verbosity (defaults to `FALSE`).

Value

Returns a `data.frame` similar to argument `x` with possibly modified content (depending on how to deal with duplicates if any).

Author(s)

Reto

annex_parse_formula *Parsing Formula*

Description

Function used to test and parse a formula used in different functions in the annex package.

Usage

```
annex_parse_formula(f, verbose = FALSE)
```

Arguments

f	object of class Formula.
verbose	logical, defaults to FALSE. Can be set to TRUE to increase verbosity.

Value

Returns a list with three components, namely vars (the variables to aggregate), time (name of the datetime variable) and group (grouping variables).

Author(s)

Reto Stauffer

See Also

annex

Examples

```
require("Formula")
annex:::annex_parse_formula(Formula(T + RH ~ datetime | study + room + home))
```

annex_prepare *Prepare Annex Data*

Description

TODO(R)

Usage

```
annex_prepare(x, config, quiet = FALSE)
```

Arguments

x	data.frame, the data itself.
config	data.frame, config information (see annex_check_config()).
quiet	logical, default FALSE. If set TRUE, messages and warnings will be suppressed.

Value

Prepared data.frame for further processing with the annex package.

Author(s)

Reto Stauffer

---

annex_read_stats *Reading Annex Statistics from XLSX*

Description

The final data set (annex statistics) are written into an XLSX file. This function allows to read one or multiple of these files into R.

Usage

```
annex_read_stats(file, raw = FALSE, validate = TRUE)
```

Arguments

file	character, name of the file(s) to be imported. Must end on XLSX (not case sensitive).
raw	logical. If FALSE a single data.frame will be returned, containing the statistics from all file(s). If set TRUE the raw information of the XLSX file(s) is returned (see Details).
validate	logical, if TRUE (default) the function first validates if the XLSX file is valid given the current version of annex.

Details

The XLSX files (if valid) contain a series of sheets, namely 'STATS', 'META-Study', 'META-Home', 'META-Room', and 'META-Variable' containing data information. This function has two different returns (depending on combine).

If raw = FALSE (default) only the 'STAT' sheet is imported. If there is more than one file the information from all files will be combined in one data.frame which is returned.

If raw = TRUE all the information from all sheets is read and stored in a named list. If there are multiple files, the content of the different sheets are combined. The return is a named list of length 5 containing a data.frame each. For usability purposes the sheet names will be modified, replacing "-" (XLSX) with "_" (names of list).

Value

An object of class `c("annex_xlsx_stats", "data.frame")` or a named list of `data.frames`. Depends argument `raw`, see section 'Details'.

Author(s)

Reto Stauffer

---

annex_room_definition *Room definition information*

Description

The template contains a series of base abbrevations allowed to define a room alongside the 'long name'. This function returns the definition as a `data.frame`.

Usage

```
annex_room_definition()
```

Value

`data.frame` with base room abbrevation, long description, plus a series of examples of valid room labels.

Author(s)

Reto Stauffer

See Also

annex_variable_definition, annex_room_definition, annex_country_definition

---

annex_stats *Calculate Statistics on Annex object*

Description

Calculate Statistics on Annex object

Usage

```
annex_stats(object, format = "wide", ..., probs = NULL)
```

Arguments

object	an object of class annex.
format	character, either "wide" (default) or "long".
...	currently unused.
probs	NULL (default; see Details) or a numeric vector of probabilities with values in [0,1] (Values will be rounded to closest 3 digits).

Details

The function allows to return the statistics in a wide format or long format. Both can be used when calling annex_write_stats(), but he long/wide format can be handy fur custom applications (e.g., plotting, ...).

Argument probs will be forwarded to the stats::quantile() function. If probs = NULL (default) the empirical quantiles will be calculated from 0 (the minimum) up to 1 (the maximum) in an interval of 0.01 (one percent steps), including quantiles 0.005, 0.025, 0.975 and 0.995. Can be specified differently by the user if needed, however, this no longer yields the standard statistics and the validation will report a problem.

Value

Returns an object of class c("annex_stats", "data_frame").

Statistics

Grouping: Statistics are calculated on different subsets (or groups), typically study, home, room, year, month, tod (time of day). However, this set can vary depending on the users function call to annex (see argument formula).

annex_stats calculates a series of data/quality flags as well as statistical measures.

Quality: quality_lower and quality_upper contain the fraction of observations (in percent) falling below the lower and upper defined threshold (see annex_variable_definition). quality_start and quality_end contain the day (date only) where the first non-missing observation was given for the current group; used to estimate Nestim (see below).

Interval: Time increments of all non-missing observations are calculated in seconds. The interval_ columns show the five digit summary plus the arithmetic mean of these intervals. interval_Median is used to calculate estimate Nestim (see below).

Nestim: Number of estimated observations (see section below) **N:** Number of non-missing observations **NAs:** Number of missing observations (NA in the data set) **Mean:**

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i$$

(arithmetic mean) **Sd:**

$$\mathrm{sd}(x) = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}\left((x_i - \bar{x})^2\right)}$$

p: Probabilites for different quantiles. `p00` represents the overall minimum, `p50` the median, `p100` the overall maximum of all non-missing values. Uses the empirical quantile function with `type = 7` (default; see `quantile`).

Note: If `N - NAs` lower than 30, both `Mean` and `Sd` will be set to `NA`!

Estimated number of observations

The value `Nestim` contains an *estimate* for the number of possible observations for a specific group. This estimate is based on the first/last date an observation was available (non-missing) as well as the `year`, `month`, and `tod`. Last but not least the `interval_Median` is used.

As an example: Imagine the statistics for temperature observations for one speicifc year and month (monthly level aggregation) with `tod = "07-23"`. The first non-missing value has been reported on the first day of the month, the last one on day 10. Given that `tod = "07-23"` covers 16 hours, this indicates that observations could be available 16 hours over 10 days = 160 hours in total. Based on the best guess for `interval_median` this allows to calculate `Nestim`. E.g., if the median interval is 300 (300 seconds = 5 minutes) this would leas to a possible number of observations `Nestim = 10 days * 16 hours per day * 3600 seonds per hour / 300 seconds = 1920`. Keep in mind that this is only an estimate or best guess!

Author(s)

Reto Stauffer

See Also

annex_stats_reshape annes_write_stats

`annex_stats_reshape` *Reshaping Annex Stats Objects*

Description

Reshaping Annex Stats Objects

Usage

```
annex_stats_reshape(x, format = NULL)
```

Arguments

`x`	object of class `annex_stats` as returned by `annex_stats()`.
`format`	`NULL` by default or one of `"long"` or `"wide"` (see Details).

Value

Returns a reshaped version of the input. If the Object provided on `x` inherits `annex_stats_wide` (wide format) the long format will be returned and vice versa if `format = NULL`. If the format is specified as either `"long"` or `"wide"` the long or wide format will be returned (possibly an unmodified version of the input if the input is already in the desired format).

Author(s)

Reto Stauffer

See Also

annex_stats

`annex_template` *Create Copy of annex Output Template*

Description

The main aim of the `annex` package is to standardize data sets for the IEA EBC Annex86 project. To create the output file, `annex_write_stats()` uses a template XLSX file (shipped with the package). This function allows to make a local copy to check the format of the template if needed.

Usage

```
annex_template(file, overwrite = FALSE)
```

Arguments

`file`	name of the file to be written, must end on xlsx (not case sensitive).
`overwrite`	logical, default is `FALSE`. Can be set to `TRUE` to overwrite an existing file (be aware of loss of data).

Value

No return.

Author(s)

Reto Stauffer

annex_validate *Validate annex Output File*

Description

Validate XLSX file created by `annex_write_stats()`. Checks if all required sheets/columns are available and that all user-modified META information has been entered correctly.

Usage

```
annex_validate(file, user, quiet = FALSE, ...)
```

Arguments

`file`	name of the file to be validated (XLSX file).
`user`	positive integer, the user identifier given by the project team.
`quiet`	logical, defaults to `FALSE`. If `TRUE`, the output will be limited.
`...`	currently unused.

Value

Some checks will cause an error and stop execution. Others will cause a message with some information on what has to be fixed to make the document valid. If the function does not stop due to an error it will return `TRUE` if the file has been validated as proper, or `FALSE` if issues have been found.

Author(s)

Reto Stauffer

annex_variable_definition *Variable definition information*

Description

The template not only contains the definition of the allowed variables, it also states whether or not additional information is required (or optional), an upper and lower bound to be considered 'valid' plus (is specified) a series of allowed units. Used to prepare the data and convert to annex standard units, quality checks, as well as validation.

Usage

```
annex_variable_definition(as_list = FALSE)
```

Arguments

as_list	logical. If FALSE (default) a data.frame will be returned, if TRUE a list (see Details).

Details

If as_list = TRUE a list of lists is returned, else a data.frame.

List: The name of the list corresponds to the name of the variable, whereas each entry contains a list with a logical flag if additional information in the META sheet is required as well as a numeric lower and upper bound which defines in which range an observation is considered to be valid. Can be NA if not specified (both or one of them). allowed_units contains NA (unspecified) or a character wich one or multiple comma separated units specifications.

If as_list = FALSE (default) the same information is returned as a data.frame containing the same information.

Value

Returns either a data.frame or list of lists which contains the allowed (defined) variables.

Author(s)

Reto Stauffer

See Also

annex_variable_definition, annex_room_definition, annex_country_definition

---

annex_write_stats *Writing Annex Stats to Disc*

Description

TODO(R)

Usage

```
annex_write_stats(x, file, user, mode = "write", ..., quiet = FALSE)
```

Arguments

x	object of class annex_stat as returned by annex_stats() (wide or long format).
file	name (or path) to the XLSX file where to store the data. Must end with xlsx (not case sensitive). See 'Details'.
user	positive integer, the user identifier given by the project team. Will be appended to the data set.

mode	character, writing mode. Can be one of "write" (default), "append" (add new data) or "update" (update existing data). See 'Mode' for more information.
...	not yet used.
quiet	logical. If set TRUE messages will be printed.

Details

This function is used to write the annex statistics - the final output - into an XLSX file. The output is based on a template file shipped with the package with a predefined format.

If the output file does not exist, the template will be copied and modified by (i) saving the data into the "STAT" sheet as well as pre-filling some additional meta sheets which have to be manually edited/entered by the user.

By default, overwrite = FALSE. If the output file already exists, the function will be terminated. However, it can be set to TRUE to allow annex_write_stats() to manipulate/overwrite the current data in that XLSX file. It tries to preserve all custom data (TODO(R): not yet implemented).

Value

No return, creates a new XLSX file (see argument file) and stores the data, or updates an existing XLSX file (see argument section 'Writing mode').

Writing mode

There are three writing modes. Warning: Depending on the mode used, existing data can get lost (i.e., removed). The following modes are available:

write: Default mode, write data into a fresh XLSX file. It is assumed that the output file does not yet exist. If it exists an error will be thrown as it is unkown if the user would like to append new data to an existing file or update (overwrite) data in an existing file.

append: Append data to an existing XLSX file. This mode expects that the file does already exist and is in the correct format (will check sheets and columns against the template). If file does not exist or the content of file does not follow the format of the template, an error will be thrown.

mode = "append" falls back to mode = "write" if the output file does not yet exist.

Else the data of x will be appended to the sheet 'STAT' and additional entries in the 'META*' sheets will be created if needed. In case the new object x contains data which are already in STAT an error will be thrown (so the new data to be appended must be unique).

update: Update the data of an existing file. **Warning:** this will delete (drop) existing data in the sheet 'STAT' and append new entries in the 'META*' sheets (won't delete existing entries). The latter could cause additional warnings when validating the file if there are 'META*' entries which are no longer needed (as the data have been deleted).

Author(s)

Reto Stauffer

check_for_allowed_rooms *Checking for Allowed Rooms*

Description

The XLSX file template.xlsx contains a series of pre-defined names for the rooms (sheet 'Definitions'). This function checks if all user defined room names are valid. Not case sensitive; will be adjusted if needed.

Usage

```
check_for_allowed_rooms(x)
```

Arguments

x	character vector with room names.

Value

Character vector (with possibly adjusted) room names, or fails. Not case sensitive for checking, but will return everything in uppercase (`toupper(x)`).

Author(s)

Reto Stauffer

See Also

annex_variable_definition, annex_room_definition, annex_country_definition

check_for_allowed_variables *Checking for Allowed Variables*

Description

The XLSX file template.xlsx contains a series of pre-defined names for the variables (sheet 'Definitions'). This function checks if all user defined variable names are valid. Not case sensitive; will be adjusted if needed.

Usage

```
check_for_allowed_variables(x)
```

Arguments

x	character vector with variable names.

Value

Character vector (with possibly adjusted) variable names, or fails.

Author(s)

Reto Stauffer

See Also

annex_variable_definition, annex_room_definition, annex_country_definition

---

demo_Bedroom *Demo data set Bedroom*

Description

One of the demo data sets used for testing and for the documentation/manuals. At tabular text file (CSV alike) containing measurement data. The data set `demo_Bedroom_config` contains the corresponding configuration used when calling `annex_prepare()`.

Author(s)

Reto Stauffer

---

demo_Bedroom_config *Dummy text 4*

Description

One of the demo data sets used for testing and for the documentation/manuals. A tabular text file (CSV alike) containing the config for the `demo_Bedroom` data set, which contains the actual measurements.

Author(s)

Reto Stauffer

demo_UIBK *Demo data set UIBK (XLSX)*

Description

One of the demo data sets used for testing and for the documentation/manuals. An XLSX file containing both, the measurements as well as a configuration to prepare the annex objects.

Author(s)

Reto Stauffer

demo_UIBK_config *Demo data set UIBK config (textfile)*

Description

One of the demo data sets used for testing and for the documentation/manuals. A tabular text file (CSV alike) containing the config information for the measurement sheet in the `demo_UIBK` data set. An alternative to the configuration contained in a dedicated sheet in the `demo_UIBK` data set.

Author(s)

Reto Stauffer

formatting *Formats a data frame in specific format*

Description

Generates a new data frame according to the formula provided.

Usage

```
formatting(
  data_frame,
  formula,
  tz = "UTC",
  format = "%Y-%m-%d %H:%M:%S",
  user = "",
  study = "",
  home = "",
  room = ""
)
```

Arguments

data_frame	a data frame
formula	a formula describing the specific layout of the data frame
tz	a time zone like "UTC", "MET", ...
format	the format in which the time stamps are given in the data frame
user	abbreviation for the name of the respective scientist
study	a natural number (entered as a character)
home	a natural number (entered as a character)
room	abbreviation for the name of the respective room

Details

The function adds columns referring to the specific season and time of day (tod) for each time stamp. Variables will be on the left side, the date terms on the right including season and tod as well as other mandatory variables (home, room, etc.). If additional variables were given, these can also be found in the output.

If a mandatory variable is not present in the output, a message is issued. Furthermore, the evaluate function of annex cannot be used unless all the required variables are available. If the formula passed contains more than just the mandatory variables, the evaluate function of annex can be used without any problems.

The output contains at least seven columns in addition to the columns for the respective variables.

Value

Returns a data frame that has a specific format according to the formula provided

See Also

base::data.frame(), stats::formula()

---

get_ID_info *Get ID information*

Description

Used for warning and error messages. Returns a printable string with IDs which have been identified.

Usage

```
get_ID_info(x, n = 5L, prefix = " ID")
```

Arguments

x	character vector.
n	integer, defaults to 5. Number of rows to be explicitly listed (if any).
prefix	character or NULL, default "ID".

Value

Character string with the information where suspicious values have been found in the XLSX file. If there are no suspicious rows, FALSE is returned.

Author(s)

Reto Stauffer

---

get_required_columns *Get required columns for warnings and infos*

Description

Different sheets contain different columns which require having the user to specify meta information or other details (no empty cells allowed). This function returns the names of these columns (as in the XLSX file) used for validation. If no definition is available, NULL is returned.

Usage

```
get_required_columns(sheet)
```

Arguments

sheet	character, name of the XLSX sheet.

Value

Returns NULL if there is no definition/no required columns, or a character vector with the exact column name as used in XLSX.

Author(s)

Reto Stauffer

`get_row_info` *Get row information*

Description

Used for warning and error messages. Returns a printable string with the information where to find these missing, invalid, ... values in the XLSX file.

Usage

```
get_row_info(x, n = 5L, offset = 1L, prefix = "  Row")
```

Arguments

`x`	integer or logical, see 'Details'.
`n`	integer, defaults to `5`. Number of rows to be explicitly listed (if any).
`offset`	integer, number of additional header lines in the XLSX sheet to correct the row indicator (defaults to `1`).
`prefix`	character or `NULL`, default `" Row"`.

Details

If the input is of class integer it is assumed that the values correspond to the observations in a `data.frame`. I.e, `1` is the first observation in the `data.frame`.

If the input is a logical vector it is assumed that all entries being `TRUE` are suspicious, and that the order corresponds to the observations in a `data.frame`.

The return value will point to these suspicous rows but in the context of an XLSX file which typically has one additional header line (thus the default `offset = 1L`).

Value

Character string with the information where suspicious values have been found in the XLSX file. If there are no suspicious rows, `FALSE` is returned.

Author(s)

Reto Stauffer

is.regular.annex *Check Regularity of an Annex Series*

Description

`is.regular` is a regular function for checking whether a series of observations has an underlying regularity or is even strictly regular. Evaulate for each group of an annex object.

Usage

```
## S3 method for class 'annex'
is.regular(x, strict = TRUE, ...)
```

Arguments

`x`	object of class `annex`.
`strict`	logical, defaults to `TRUE`. If `FALSE`, regularity (but not strict regularity) will be checked.
`...`	currently unused.

Value

Returns a named logical vector where the name is a combination of the grouping (study, home, room), the content the result of checking regularity.

Author(s)

Reto stauffer

ISO3166 *Country codes (ISO 3166) alpha-2 and alpha-3*

Description

List of countries with ISO-2 and ISO-3 abbrevations. Used to validate the final XLSX file.

Value

Returns a `data.frame` containing the `country` name alongside `ISO2` and `ISO3` short name (ISO 3116 alpha-2 and alpha-3 standard).

Author(s)

Reto Stauffer

References

IBAN (2023). COUNTRY CODES ALPHA-2 & ALPHA-3, https://www.iban.com/country-codes, accessed 2023-02-11.

`plot.annex` *Standard plot for annex objects*

Description

TODO(R)

Usage

```
## S3 method for class 'annex'
plot(x, bygroup = FALSE, start = NULL, end = NULL, ...)
```

Arguments

`x`	an object of class `annex`.
`bygroup`	logical, by default the subplots are build up on different variables. If `TRUE`, all varables from one group will be plotted in one subplot.
`start`	the start time of the period of interest.
`end`	the end time of the period of interest.
`...`	currently unused.

Author(s)

Reto Stauffer

`plot.annex_stats` *Standard plot for annex_stats objects*

Description

Experimental function for objects of class `annex_stats`.

Usage

```
## S3 method for class 'annex_stats'
plot(
  x,
  tod = c("07-23", "23-07", "all"),
  by = c("year", "yearmon"),
  ncol = 1L,
  ask = NULL,
  ...
)
```

Arguments

x	an object of class annex_stats
tod	time of day
by	should the statistics be plotted across years or year + month
ncol	number of columns in plot
ask	one of NULL, TRUE, or FALSE. Auto-detected if NULL
...	currently unused.

Author(s)

Reto Stauffer

---

summary.annex *Annex Summary*

Description

Numeric summary of an annex object.

Usage

```
## S3 method for class 'annex'
summary(object, type = "default", ...)
```

Arguments

object	an object of class annex.
type	character, one of "default" (default) or "statistic". If type = "statistics" the result of annex_stats(object) will be printed.
...	currently unused.

Value

Returns NULL (invisible).

Author(s)

Reto stauffer

Index