Package 'melidosData'

May 8, 2026

Title Load Data from the MeLiDos Field Study

Version 1.0.6

Description In the MeLiDos field study, personal light exposure data were collected in 9 sites, 7 countries, and 196 participants following the Guidolin et al. (2024) <doi:10.1186/s12889-024-20206-4> protocol. Data originate from wearable devices collecting personal light exposure at the eye level, chest, and the wrist. Questionnaires were collected via 'REDCap' and contain demographic information as well as chronotype, current conditions, sleep diaries, wear logs, and many more. This package makes loading the data from the respective repositories (<https://github.com/MeLiDosProject>) into R a breeze. It further contains some quality of life functions for label handling and data from 'REDCap'.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

Depends R (>= 4.1.0)

LazyData true

Imports dplyr, hms, lubridate, purrr, rlang, stringr, tibble

Suggests gt, LightLogR, testthat (>= 3.0.0), vroom

Config/testthat/edition 3

URL https://melidosproject.github.io/melidosData/,
https://github.com/MeLiDosProject/melidosData,
https://github.com/MeLiDosProject

BugReports https://github.com/MeLiDosProject/melidosData/issues

NeedsCompilation no

Author Johannes Zauner [aut, cre] (ORCID:
<https://orcid.org/0000-0003-2171-4566>),
Manuel Spitschan [aut] (ORCID: <https://orcid.org/0000-0002-8572-9268>),
European Partnership on Metrology [fnd] (The project (22NRM05 MeLiDos)
has received funding from the European Partnership on Metrology,
co-financed by the European Union's Horizon Europe Research and
Innovation Programme, EURAMET, and the Participating States. Views

and opinions expressed are those of the authors and do not
necessarily reflect those of the European Union or EURAMET.),
Translational Sensory and Circadian Neuroscience Unit
(MPS/TUM/TUMCREATE) [cph] (URL: www.tscnlab.org)

Maintainer Johannes Zauner <johannes.zauner@tum.de>

Repository CRAN

Date/Publication 2026-04-22 14:00:08 UTC

Contents

`add_label` *Add a* `label` *attribute to a vector*

Description

Add a `label` attribute to a vector

Usage

```
add_label(data, label)
```

Arguments

`data`	Vector-like object.
`label`	Label text.

Value

`data` with a `label` attribute.

Examples

```
add_label(1:3, "Example variable")
```

---

add_labels *Add labels to selected data frame columns*

Description

Add labels to selected data frame columns

Usage

```
add_labels(data, labels)
```

Arguments

`data`	A data frame.
`labels`	Named character vector or named list of labels.

Value

A tibble with column-level `label` attributes where names matched.

Examples

```
dat <- data.frame(a = 1:2, b = 3:4)
labelled <- add_labels(dat, c(a = "Column A"))
attr(labelled$a, "label")
attr(labelled$b, "label")
```

---

codebook_strings_to_ignore *Codebook strings to ignore, as it contains HTML formatting*

Description

Codebook strings to ignore, as it contains HTML formatting

Usage

```
codebook_strings_to_ignore
```

Format

A data frame.

`extract_labels` *Extract labels from a list of objects*

Description

Retrieves the `"label"` attribute from each element in a list and returns a character vector of labels.

Usage

```
extract_labels(data)
```

Arguments

`data`	A list of objects. Each element is expected to have a `"label"` attribute (e.g., created via `attr(x, "label") <- "some label"`).

Value

A character vector containing the `"label"` attribute of each element in `data`. If an element does not have a `"label"` attribute, `NA_character_` is returned for that element.

Examples

```
# Create example data with labels
x1 <- 1:3
attr(x1, "label") <- "First variable"

x2 <- 4:6
attr(x2, "label") <- "Second variable"

x3 <- 7:9
# No label set for x3

data <- list(x1, x2, x3)

# Extract labels
extract_labels(data)

# Works for data frames as well
data <- data.frame (x1, x2, x3)
extract_labels(data)
```

`flatten_data` *Flatten multi-site MeLiDos data into one table*

Description

`flatten_data()` combines the named list returned by `load_data()` into one tibble and keeps site provenance in a `site` column.

Usage

```
flatten_data(melidos_data, tz = "UTC", label_from = 1)
```

Arguments

`melidos_data`	A list returned by `load_data()` for multiple sites.
`tz`	Time zone to enforce for all `POSIXct` columns.
`label_from`	indice of the dataset that is used to apply labels to the output.

Details

If date-time columns are present (`POSIXct`), their timezone is overwritten using `lubridate::force_tz()`.

Value

A tibble with all rows stacked and a `site` column.

Examples

```
example_multi_site <- structure(
  list(
    TUM = data.frame(id = 1, bedtime = as.POSIXct("2024-01-01 22:00:00", tz = "UTC")),
    UCR = data.frame(id = 2, bedtime = as.POSIXct("2024-01-02 22:30:00", tz = "UTC"))
  ),
  class = c("melidos_data", "list")
)

flatten_data(example_multi_site, tz = "Europe/Berlin")
```

load_data *Load MeLiDos datasets from project repositories*

Description

load_data() is the main entry point of the package. It downloads one modality from one or more MeLiDos sites and returns either a single data frame (one site) or a named list with class "melidos_data" (multiple sites).

Usage

```
load_data(
 modality = c("light_glasses", "light_chest", "light_wrist", "light_glasses_1minute",
  "light_chest_1minute", "light_wrist_1minute", "acceptability", "ase", "chronotype",
   "demographics", "evaluation", "health", "leba", "trial_times", "vlsq8",
  "currentconditions", "exercisediary", "experiencelog", "lightexposurediary",
   "sleepdiaries", "wearlog", "wellbeingdiary"),
 site = c("all", "BAUA", "FUSPCEU", "IZTECH", "KNUST", "MPI", "RISE", "THUAS", "TUM",
   "UCR")
)
```

Arguments

modality	Dataset to load.
site	Site(s) to load. Use "all" for all available sites.

Details

Use flatten_data() to stack multi-site results into one tibble with a site column.

See the README of the package for a description of sites and modalities.

Value

A data frame when one site is selected, or a melidos_data list for multiple sites.

Source

https://github.com/MeLiDosProject

Examples

```
# load one questionnaire modality for two sites
sleep_all <- load_data("sleepdiaries", site = c("TUM", "RISE"))

# flatten to a single tibble with a site column
sleep_flat <- flatten_data(sleep_all, tz = "UTC")
head(sleep_flat)
```

```
# load one site only (returns a data frame)
sleep_tum <- load_data("sleepdiaries", site = "TUM")
```

`melidos_cities` *MeLiDos site cities*

Description

Named character vector of city names keyed by study site abbreviation.

Usage

```
melidos_cities
```

Format

A named character vector.

Source

MeLiDos study metadata.

`melidos_colors` *MeLiDos site colors*

Description

Named character vector of hex colors keyed by study site abbreviation.

Usage

```
melidos_colors
```

Format

A named character vector.

Source

MeLiDos study metadata.

melidos_coordinates *MeLiDos site coordinates*

Description

Named list of latitude/longitude numeric pairs keyed by study site abbreviation.

Usage

```
melidos_coordinates
```

Format

A named list.

Source

MeLiDos study metadata.

melidos_countries *MeLiDos site countries*

Description

Named character vector of country names keyed by study site abbreviation.

Usage

```
melidos_countries
```

Format

A named character vector.

Source

MeLiDos study metadata.

melidos_tzs *MeLiDos site time zones*

Description

Named character vector of Olson time zone identifiers keyed by study site abbreviation.

Usage

```
melidos_tzs
```

Format

A named character vector.

Source

MeLiDos study metadata.

nighttime_switch *Summarise times across midnight*

Description

Converts times before `cutoff` to the next day before applying `.fun`, which helps summarise nighttime values spanning midnight (for example, median sleep time).

Usage

```
nighttime_switch(
  datetime,
  .fun = stats::median,
  cutoff = 12 * 60 * 60,
  hms = TRUE
)
```

Arguments

`datetime`	A date-time vector coercible to `POSIXct`.
`.fun`	Summary function applied after the date shift.
`cutoff`	Cutoff in seconds since midnight. Values below cutoff are treated as belonging to the next day.
`hms`	Logical; if `TRUE`, return an `hms` object.

Value

A summarised time as hms (default) or date-time.

Examples

```
x <- as.POSIXct(c("2024-01-01 23:00:00", "2024-01-02 01:00:00"), tz = "UTC")
nighttime_switch(x)
```

REDCap_attention_check

Evaluate a REDCap attention check column

Description

Replaces a column with a logical pass/fail value based on membership in condition, sets a label attribute, and moves the column to the end.

Usage

```
REDCap_attention_check(
  data,
  check.column,
  condition,
  label = "Attention check successful"
)
```

Arguments

data	A data frame.
check.column	Unquoted column to evaluate.
condition	Vector of accepted responses.
label	Label to attach to the resulting logical column.

Value

data with an updated attention check column.

Examples

```
dat <- data.frame(attention = c("A", "B", "A"))
REDCap_attention_check(dat, attention, condition = "A")
```

REDCap_codebook_prepare *Prepare a REDCap codebook for downstream processing*

Description

Cleans HTML formatting from codebook labels and optionally filters the codebook to selected REDCap forms.

Usage

```
REDCap_codebook_prepare(
  codebook,
  strings_to_ignore = codebook_strings_to_ignore,
  form.filter = NULL,
  field.label = `Field Label`,
  form.col = `Form Name`
)
```

Arguments

codebook	A REDCap data dictionary as a data frame.
strings_to_ignore	A regular expression used in stringr::str_remove_all() to remove formatting fragments from field labels.
form.filter	Optional character vector of REDCap form names to keep.
field.label	Unquoted column containing question labels.
form.col	Unquoted column containing the form name (for filtering)

Value

A cleaned codebook tibble/data frame.

Examples

```
codebook_path <- system.file("ext", "DataDictionary_sleepdiary.csv",
  package = "melidosData"
)
codebook <- utils::read.csv(codebook_path, check.names = FALSE)
cleaned <- REDCap_codebook_prepare(codebook)
cleaned$`Field Label`
#compate the original one
codebook$`Field Label`
```

REDCap_coltype_check *Check data column classes against REDCap expectations*

Description

Uses REDCap codebook metadata to infer expected classes and compares these to classes in `data`.

Usage

```
REDCap_coltype_check(
  codebook,
  indicator_POSIXct = "datetime_dmy",
  indicator_date = "Date",
  indicator_time = "time",
  indicator_logical = "yesno",
  indicator_numeric.val_col = c("number", "integer"),
  indicator_numeric.type_col = c("radio", "dropdown"),
  label_col = `Field Label`,
  name_col = `Variable / Field Name`,
  type_col = `Field Type`,
  val_col = `Text Validation Type OR Show Slider Number`,
  data
)
```

Arguments

`codebook`	REDCap data dictionary.
`indicator_POSIXct`	Indicator in `val_col` identifying datetime fields.
`indicator_date`	Pattern used in labels to identify date variables.
`indicator_time`	Indicator in `val_col` identifying time-only fields.
`indicator_logical`	Indicator in `type_col` identifying logical fields.
`indicator_numeric.val_col`	Indicators in `val_col` for numeric fields.
`indicator_numeric.type_col`	Indicators in `type_col` for numeric fields.
`label_col`	Unquoted codebook label column.
`name_col`	Unquoted codebook variable-name column.
`type_col`	Unquoted codebook field-type column.
`val_col`	Unquoted codebook validation/type-hint column.
`data`	Data frame to validate.

Value

A list with ok, summary, and per-column details.

Examples

```
library(gt)
dict_path <- system.file("ext", "DataDictionary_sleepdiary.csv",
  package = "melidosData"
)
dict <- utils::read.csv(dict_path, check.names = FALSE)

coltype_check <- REDCap_coltype_check(dict, data = REDCap_example_sleep)
coltype_check$ok
coltype_check$summary
coltype_check$details |> gt()

dict_path <- system.file("ext", "DataDictionary_chronotype.csv",
  package = "melidosData"
)
dict <- utils::read.csv(dict_path, check.names = FALSE)
dict <- REDCap_codebook_prepare(dict, form.filter = "mctq")

coltype_check <- REDCap_coltype_check(dict, data = REDCap_example_chronotype)
coltype_check$ok
coltype_check$summary
coltype_check$details
```

REDCap_col_labels *Add variable labels from a REDCap codebook*

Description

Sets the label attribute on matching columns in data.

Usage

```
REDCap_col_labels(
  data,
  lookup,
  var_col = `Variable / Field Name`,
  label_col = `Field Label`,
  warn = TRUE
)
```

Arguments

data	A data frame.
lookup	A data frame with variable names and labels.

var_col	Unquoted column in lookup containing variable names.
label_col	Unquoted column in lookup containing human-readable labels.
warn	Logical; warn when labels are provided for variables not present in data.

Value

data with label attributes added.

Examples

```
dict_path <- system.file("ext", "DataDictionary_chronotype.csv",
  package = "melidosData"
)
dict <- utils::read.csv(dict_path, check.names = FALSE)
dict <- REDCap_codebook_prepare(dict,  form.filter = "mctq")
labelled <- REDCap_col_labels(REDCap_example_chronotype, dict)
attr(labelled[[5]], "label")
```

REDCap_example_chronotype

Example REDCap chronotype data

Description

Example export containing chronotype questionnaire responses.

Usage

```
REDCap_example_chronotype
```

Format

A data frame.

Source

inst/ext/example_chronotype.csv.

REDCap_example_sleep *Example REDCap sleep diary data*

Description

Example export containing sleep diary responses.

Usage

```
REDCap_example_sleep
```

Format

A data frame.

Source

`inst/ext/example_sleepdiary.csv`.

REDCap_factors *Convert REDCap choice fields to factors*

Description

Applies factor levels and labels using REDCap dictionary definitions for `radio` and `checkbox` fields (configurable via `radio_value`).

Usage

```
REDCap_factors(
  data,
  lookup,
  var_col = `Variable / Field Name`,
  type_col = `Field Type`,
  levels_col = `Choices, Calculations, OR Slider Labels`,
  radio_value = c("checkbox", "radio"),
  warn = TRUE
)
```

Arguments

data	Data frame containing REDCap records.
lookup	Data frame containing variable metadata.
var_col	Unquoted column in lookup with variable names.
type_col	Unquoted column in lookup with REDCap field types.
levels_col	Unquoted column in lookup with coded levels ("1, Yes | 0, No" format).
radio_value	Character vector of field types to convert.
warn	Logical; warn when variables are in lookup but absent from data.

Value

data with selected columns converted to factors.

Examples

```
dict_path <- system.file("ext", "DataDictionary_chronotype.csv",
  package = "melidosData"
)
dict <- utils::read.csv(dict_path, check.names = FALSE)
dict <- REDCap_codebook_prepare(dict, form.filter = "mctq")

chronotype_with_factors <-
REDCap_factors(
  data = REDCap_example_chronotype,
  lookup = dict
)

chronotype_with_factors$mctq_work_travel
#original:
REDCap_example_chronotype$mctq_work_travel
```

Index