Package ‘REDCapCAST’

July 4, 2023

Title  REDCap Castellated Data Handling
Version  23.6.2
Description  Forked from 'REDCapRITS' by Paul Egeler and Spectrum Health. See <https://github.com/SpectrumHealthResearch/REDCapRITS>. Handles castellated datasets from 'REDCap' projects with repeating instruments. Assists in casting tidy tables from raw 'REDCap' data exports for each repeated instrument. Keeps a focused data export approach, by allowing to only export required data from the database. 'REDCap' (Research Electronic Data Capture) is a secure, web-based software platform designed to support data capture for research studies, providing 1) an intuitive interface for validated data capture; 2) audit trails for tracking data manipulation and export procedures; 3) automated export procedures for seamless data downloads to common statistical packages; and 4) procedures for data integration and interoperability with external sources (Harris et al (2009) <doi:10.1016/j.jbi.2008.08.010>, Harris et al (2019) <doi:10.1016/j.jbi.2019.103208>).

Depends  R (>= 3.4.0)
Suggests  spelling, RCurl, httr, jsonlite, testthat, Hmisc, readr, covr, knitr, rmarkdown, gt, keyring
License  GPL (>= 3)
Encoding  UTF-8
LazyData  true
RoxygenNote  7.2.3
URL  https://github.com/agdamsbo/REDCapCAST,
     https://agdamsbo.github.io/REDCapCAST/
BugReports  https://github.com/agdamsbo/REDCapCAST/issues
Imports  dplyr, REDCapR, tidyR, tidyselect
Collate  'utils.r' 'process_user_input.r' 'REDCap_split.r' 'ds2dd.R'
        'read_redcap_tables.R' 'redcap_wider.R' 'redcapcast_data.R'
        'redcapcast_meta.R'
Language  en-US
### Description

Stepwise removal on non-alphanumeric characters, trailing white space, substitutes spaces for underscores and converts to lower case. Trying to make up for different naming conventions.

### Usage

```r
clean_redcap_name(x)
```

### Arguments

- `x` vector or data frame for cleaning

### Value

vector or data frame, same format as input
**d2w**  
*Convert single digits to words*

**Description**  
Convert single digits to words

**Usage**  
d2w(x, lang = "en", neutrum = FALSE, everything = FALSE)

**Arguments**  
- **x**: data. Handle vectors, data.frames and lists  
- **lang**: language. Danish (da) and English (en). Default is "en"  
- **neutrum**: for numbers depending on counted word  
- **everything**: flag to also split numbers >9 to single digits

**Value**  
returns characters in same format as input

**Examples**  
d2w(c(2:8,21))  
d2w(data.frame(2:7,3:8,1),lang="da",neutrum=TRUE)

## If everything=T, also larger numbers are reduced.  
## Elements in the list are same length as input  
d2w(list(2:8,c(2,6,4,23),2), everything=TRUE)

---

**ds2dd**  
*Data set to data dictionary function*

**Description**  
Migrated from stRoke ds2dd(). Fits better with the functionality of `REDCapCAST`
Usage

```r
ds2dd(
  ds,
  record.id = "record_id",
  form.name = "basis",
  field.type = "text",
  field.label = NULL,
  include.column.names = FALSE,
  metadata = names(redcapcast_meta)
)
```

Arguments

- `ds`: data set
- `record.id`: name or column number of id variable, moved to first row of data dictionary, character of integer. Default is "record_id".
- `form.name`: vector of form names, character string, length 1 or length equal to number of variables. Default is "basis".
- `field.type`: vector of field types, character string, length 1 or length equal to number of variables. Default is "text".
- `field.label`: vector of form names, character string, length 1 or length equal to number of variables. Default is NULL and is then identical to field names.
- `include.column.names`: Flag to give detailed output including new column names for original data set for upload.
- `metadata`: Metadata column names. Default is the included REDCapCAST::redcapcast_data.

Value

data.frame or list of data.frame and vector

Examples

```r
redcapcast_data$record_id <- seq_len(nrow(redcapcast_data))
ds2dd(redcapcast_data, include.column.names=TRUE)
```

Description

Extracts limited metadata for variables in a dataset

Usage

`focused_metadata(metadata, vars_in_data)`
**match_fields_to_form**

**Arguments**

- **metadata**: A dataframe containing metadata
- **vars_in_data**: Vector of variable names in the dataset

**Value**

A dataframe containing metadata for the variables in the dataset

**Description**

Match fields to forms

**Usage**

```r
match_fields_to_form(metadata, vars_in_data)
```

**Arguments**

- **metadata**: A data frame containing field names and form names
- **vars_in_data**: A character vector of variable names

**Value**

A data frame containing field names and form names

---

**read_redcap_tables**

*Download REDCap data*

**Description**

Implementation of REDCap_split with a focused data acquisition approach using REDCapR::redcap_read nad only downloading specified fields, forms and/or events using the built-in focused_metadata in- cluding some clean-up. Works with longitudinal projects with repeating instruments.
Usage

```r
read_redcap_tables(
  uri,
  token,
  records = NULL,
  fields = NULL,
  events = NULL,
  forms = NULL,
  raw_or_label = "label",
  split_forms = "all",
  generics = c("record_id", "redcap_event_name", "redcap_repeat_instrument",
               "redcap_repeat_instance")
)
```

Arguments

- `uri` REDCap database uri
- `token` API token
- `records` records to download
- `fields` fields to download
- `events` events to download
- `forms` forms to download
- `raw_or_label` raw or label tags
- `split_forms` Whether to split "repeating" or "all" forms, default is all.
- `generics` vector of auto-generated generic variable names to ignore when discarding empty rows

Value

- list of instruments

Examples

```
# Examples will be provided later
```

redcapcast_data  

**Data set for demonstration**

Description

This is a small dataset from a REDCap database for demonstrational purposes. Contains only synthetic data.
**Usage**

```r
data(redcapcast_data)
```

**Format**

A data frame with 22 variables:

- `record_id` ID, numeric
- `redcap_event_name` Event name, character
- `redcap_repeat_instrument` Repeat instrument, character
- `redcap_repeat_instance` Repeat instance, numeric
- `cpr` CPR number, character
- `inclusion` Inclusion date, date
- `dob` Date of birth, date
- `age` Age decimal, numeric
- `age_integer` Age integer, numeric
- `sex` Legal sex, character
- `cohabitation` Cohabitation status, character
- `hypertension` Hypertension, character
- `diabetes` diabetes, character
- `region` region, character
- `baseline_data_start_complete` Completed, character
- `mrs_assessed` mRS Assessed, character
- `mrs_date` Assessment date, date
- `mrs_score` Score, numeric
- `mrs_complete` Complete, numeric
- `event_date` Event date, date
- `event_type` Event type, character
- `new_event_complete` Completed, character

---

**Description**

This metadata dataset from a REDCap database is for demonstrational purposes.

**Usage**

```r
data(redcapcast_meta)
```
**Format**

A data frame with 22 variables:

- **field_name** field_name, character
- **form_name** form_name, character
- **section_header** section_header, character
- **field_type** field_type, character
- **field_label** field_label, character
- **select_choices_or_calculations** select_choices_or_calculations, character
- **field_note** field_note, character
- **text_validation_type_or_show_slider_number** text_validation_type_or_show_slider_number, character
- **text_validation_min** text_validation_min, character
- **text_validation_max** text_validation_max, character
- **identifier** identifier, character
- **branching_logic** branching_logic, character
- **required_field** required_field, character
- **custom_alignment** custom_alignment, character
- **question_number** question_number, character
- **matrix_group_name** matrix_group_name, character
- **matrix_ranking** matrix_ranking, character
- **field_annotation** field_annotation, character

---

**REDCap_split**  
*Split REDCap repeating instruments table into multiple tables*

**Description**

This will take output from a REDCap export and split it into a base table and child tables for each repeating instrument. Metadata is used to determine which fields should be included in each resultant table.

**Usage**

```r
REDCap_split(
  records,
  metadata,
  primary_table_name = "",
  forms = c("repeating", "all")
)
```
Arguments

records  Exported project records. May be a data.frame, response, or character vector containing JSON from an API call.
metadata  Project metadata (the data dictionary). May be a data.frame, response, or character vector containing JSON from an API call.
primary_table_name  Name given to the list element for the primary output table (as described in README.md). Ignored if forms = 'all'.
forms  Indicate whether to create separate tables for repeating instruments only or for all forms.

Value

A list of "data.frame"s. The number of tables will differ depending on the forms option selected.

- 'repeating': one base table and one or more tables for each repeating instrument.
- 'all': a data.frame for each instrument, regardless of whether it is a repeating instrument or not.

Author(s)

Paul W. Egeler, M.S., GStat

Examples

```r
## Not run:
# Using an API call -------------------------------------------------------
library(RCurl)

# Get the records
records <- postForm(
  uri = api_url,  # Supply your site-specific URI
token = api_token,  # Supply your own API token
  content = 'record',
  format = 'json',
  returnFormat = 'json'
)

# Get the metadata
metadata <- postForm(
  uri = api_url,  # Supply your site-specific URI
token = api_token,  # Supply your own API token
  content = 'metadata',
  format = 'json'
)

# Convert exported JSON strings into a list of data.frames
REDCapRITS::REDCap_split(records, metadata)
```
# Using a raw data export -----------------------------------------------

# Get the records
records <- read.csv("/path/to/data/ExampleProject_DATA_2018-06-03_1700.csv")

# Get the metadata
metadata <- read.csv("/path/to/data/ExampleProject_DataDictionary_2018-06-03.csv")

# Split the tables
REDCapRITS::REDCap_split(records, metadata)

# In conjunction with the R export script ---------------------------------

# You must set the working directory first since the REDCap data export 
# script contains relative file references.
old <- getwd()
setwd("/path/to/data/")

# Run the data export script supplied by REDCap.
# This will create a data.frame of your records called 'data'
source("ExampleProject_R_2018-06-03_1700.r")

# Get the metadata
metadata <- read.csv("ExampleProject_DataDictionary_2018-06-03.csv")

# Split the tables
REDCapRITS::REDCap_split(data, metadata)
setwd(old)

## End(Not run)

---

**Redcap Wider**

### Description

Converts a list of REDCap data frames from long to wide format. Handles longitudinal projects, but not yet repeated instruments.

### Usage

```r
redcap_wider(list, 
  event.glue = "{.value}_{redcap_event_name}",
  inst.glue = "{.value}_{redcap_repeat_instance}"
)
```
Arguments

list  A list of data frames.
event.glue A dplyr::glue string for repeated events naming
inst.glue A dplyr::glue string for repeated instruments naming

Value

The list of data frames in wide format.

Examples

```r
list <- list(data.frame(record_id = c(1,2,1,2),
                      redcap_event_name = c("baseline", "baseline", "followup", "followup"),
                      age = c(25,26,27,28)),
             data.frame(record_id = c(1,2),
                        redcap_event_name = c("baseline", "baseline"),
                        gender = c("male", "female")))
redcap_wider(list)
```

Description

Removing empty rows

Usage

```r
sanitize_split(
  l,
  generic.names = c("record_id", "redcap_event_name", "redcap_repeat_instrument",
                      "redcap_repeat_instance")
)
```

Arguments

1  A list of data frames.

generic.names  A vector of generic names to be excluded.

Value

A list of data frames with generic names excluded.
split_non_repeating_forms

Split a data frame into separate tables for each form

Description
Split a data frame into separate tables for each form

Usage
split_non_repeating_forms(table, universal_fields, fields)

Arguments
- table: A data frame
- universal_fields: A character vector of fields that should be included in every table
- fields: A two-column matrix containing the names of fields that should be included in each form

Value
A list of data frames, one for each non-repeating form

Examples
# Create a table
table <- data.frame(
  id = c(1, 2, 3, 4, 5),
  form_a_name = c("John", "Alice", "Bob", "Eve", "Mallory"),
  form_a_age = c(25, 30, 25, 15, 20),
  form_b_name = c("John", "Alice", "Bob", "Eve", "Mallory"),
  form_b_gender = c("M", "F", "M", "F", "F")
)

# Create the universal fields
universal_fields <- c("id")

# Create the fields
fields <- matrix(
  c("form_a_name", "form_a",
    "form_a_age", "form_a",
    "form_b_name", "form_b",
    "form_b_gender", "form_b"),
  ncol = 2, byrow = TRUE
)

# Split the table
split_non_repeating_forms(table, universal_fields, fields)
strsplitx is an extended string splitting function that can be used as a substitute of the base function. Its main feature is the ability to split around a defined delimiter, as shown in the example below.

### Usage

```r
strsplitx(x, split, type = "classic", perl = FALSE, ...)
```

### Arguments

- **x**: data
- **split**: delimiter
- **type**: Split type. Can be c("classic", "before", "after", "around")
- **perl**: perl param from strsplit()
- **...**: additional parameters are passed to base strsplit handling splits

### Value

A list

### Examples

```r
test <- c("12 months follow-up", "3 steps", "mRS 6 weeks", "Counting to 231 now")
strsplitx(test, "[0-9]", type="around")
```
Index

* datasets
  redcapcast_data, 6
  redcapcast_meta, 7

clean_redcap_name, 2

d2w, 3
ds2dd, 3

focused_metadata, 4

match_fields_to_form, 5

read_redcap_tables, 5
REDCap_split, 8
redcap_wider, 10
redcapcast_data, 6
redcapcast_meta, 7

sanitize_split, 11
split_non_repeating_forms, 12
strsplitx, 13