

Package ‘REDCapR’

January 7, 2014

Title Interaction between R and REDCap

Description Encapsulates functions to streamline calls from R

Version 0.1-8

Date 2014-01-07

Author Will Beasley, David Bard, Thomas Wilson

Maintainer 'Will Beasley' <wibeasley@hotmail.com>

URL <http://ouhsc.edu/bbmc/>

Depends R(>= 3.0.0),stats

Imports methods,plyr,RCurl

Suggests devtools,ggplot2,knitr,testit,testthat

License GPL-2

LazyData TRUE

VignetteBuilder knitr

R topics documented:

REDCapR	1
redcap_read	2
validate_for_write	4

Index	5
--------------	----------

REDCapR	<i>REDCapR</i>
---------	----------------

Description

REDCapR

redcap_read

*Read records from a REDCap project.***Description**

This function uses REDCap’s **API** to select and return data.

Usage

```
redcap_read(redcap_uri, token, records = NULL, records_collapsed = NULL,
  fields = NULL, fields_collapsed = NULL,
  export_data_access_groups = FALSE, raw_or_label = "raw", verbose = TRUE,
  cert_location = NULL)
```

Arguments

redcap_uri	The URI of the REDCap project. Required.
token	The user-specific string that serves as the password for a project. Required.
records	An array, where each element corresponds to the ID of a desired record. Optional.
records_collapsed	A single string, where the desired ID values are separated by commas. Optional.
fields	An array, where each element corresponds a desired project field. Optional.
fields_collapsed	A single string, where the desired field names are separated by commas. Optional.
export_data_access_groups	A boolean value that specifies whether or not to export the “redcap_data_access_group” field when data access groups are utilized in the project. Default is FALSE. See the details below.
raw_or_label	A string (either 'raw' or 'label' that specifies whether to export the raw coded values or the labels for the options of multiple choice fields. Default is 'raw'.
verbose	A boolean value indicating if messages should be printed to the R console during the operation. Optional.
cert_location	If present, this string should point to the location of the cert files required for SSL verification. If the value is missing or NULL, the server’s identity will be verified using a recent CA bundle from the cURL website . See the details below. Optional.

Details

I like how **PyCap** creates a ‘project’ object with methods that read and write from REDCap. However this isn’t a style that R clients typically use. I like the logic that it’s associated with a particular REDCap project that shouldn’t change between calls. As a compromise, I think I’ll wrap the uri, token, and cert location into a single S4 object that’s passed to these methods. It will make these calls take less space.

The ‘REDCapR’ package includes a recent version of the **Bundle of CA Root Certificates** from the official **cURL site**. This version is used by default, unless the ‘cert_location’ parameter is given another location.

If you do not pass in this `export_data_access_groups` value, it will default to `FALSE`. The following is from the API help page for version 5.2.3: This flag is only viable if the user whose token is being used to make the API request is **not** in a data access group. If the user is in a group, then this flag will revert to its default value.

Value

Currently, a list is returned with the following elements,

1. `data`: an R data frame of the desired records and columns.
2. `raw_csv`: the text of comma separated values returned by REDCap through `RCurl`.
3. `records_collapsed`: the desired records IDs, collapsed into a single string, separated by commas.
4. `fields_collapsed`: the desired field names, collapsed into a single string, separated by commas.
5. `elapsed_seconds`: the duration of the function.
6. `status_message`: a boolean value indicating if the operation was apparently successful.

Author(s)

Will Beasley

References

The official documentation can be found on the ‘API Examples’ page on the REDCap wiki (<https://iwg.devguard.com/trac/redcap/wiki/ApiExamples>). A user account is required.

The official [cURL site](#) discusses the process of using SSL to verify the server being connected to.

Examples

```
## Not run:
library(REDCapR) #Load the package into the current R session.
uri <- "https://bbmc.ouhsc.edu/redcap/api/"
token <- "9A81268476645C4E5F03428B8AC3AA7B"
#Return all records and all variables.
ds_all_rows_all_fields <- redcap_read(redcap_uri=uri, token=token)$data

#Return only records with IDs of 1 and 3
desired_records_v1 <- c(1, 3)
ds_some_rows_v1 <- redcap_read(
  redcap_uri=uri,
  token=token,
  records=desired_records_v1
)$data

#Return only the fields recordid, first_name, and age
desired_fields_v1 <- c("recordid", "first_name", "age")
ds_some_fields_v1 <- redcap_read(
  redcap_uri=uri,
  token=token,
  fields=desired_fields_v1
)$data

## End(Not run)
```

validate_for_write	<i>Inspect a data.frame to anticipate problems before writing to a REDCap project.</i>
--------------------	--

Description

This set of functions inspect a `data.frame` to anticipate problems before writing with REDCap's **API**.

Usage

```
validate_for_write( df )

validate_no_logical( df )

validate_no_uppercase( df )
```

Arguments

<code>df</code>	The <code>data.frame</code> containing the dataset used to update the REDCap project. Required.
-----------------	---

Details

All functions listed in the Usage section above inspect a specific aspect of the dataset. The `validate_for_read()` function executes all these individual validation checks. It allows the client to check everything with one call.

Value

A `data.frame`, where each potential violation is a row. The two columns are:

1. `field_name`: The name of the data.frame that might cause problems during the upload.
2. `field_index`: The position of the field. (For example, a value of '1' indicates the first column, while a '3' indicates the third column.)
3. `concern`: A description of the problem potentially caused by the field.
4. `suggestion`: A *potential* solution to the concern.

Author(s)

Will Beasley

Examples

```
df <- data.frame(
  recordid = 1:4,
  flag_logical = c(TRUE, TRUE, FALSE, TRUE),
  flag_uppercase = c(4, 6, 8, 2)
)
validate_for_write(df = df)
```

Index

redcap_read, [2](#)
REDCapR, [1](#)
REDCapR-package (REDCapR), [1](#)

validate_for_write, [4](#)
validate_no_logical
 (validate_for_write), [4](#)
validate_no_uppercase
 (validate_for_write), [4](#)