Package: qualtRics (via r-universe)

November 10, 2024

Type Package

Title Download 'Qualtrics' Survey Data

Version 3.2.1.9000

Description Provides functions to access survey results directly into R using the 'Qualtrics' API. 'Qualtrics' <https://www.qualtrics.com/about/> is an online survey and data collection software platform. See <https://api.qualtrics.com/> for more information about the 'Qualtrics' API. This package is community-maintained and is not officially supported by 'Qualtrics'.

License MIT + file LICENSE

URL https://docs.ropensci.org/qualtRics/, https://github.com/ropensci/qualtRics

BugReports https://github.com/ropensci/qualtRics/issues

Imports archive, cli, dplyr (>= 1.0), fs, glue, httr, jsonlite, lifecycle, lubridate, purrr, readr, rlang, sjlabelled, stringr, tibble, tidyr, withr

Suggests covr, knitr, rmarkdown, testthat (>= 3.0.0), vcr (>= 1.2.0), webmockr

VignetteBuilder knitr

Encoding UTF-8

LazyData FALSE

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Config/testthat/edition 3

Config/pak/sysreqs make libarchive-dev libicu-dev libssl-dev libx11-dev

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

RemoteUrl https://github.com/ropensci/qualtRics

RemoteRef main

RemoteSha 97f1d1a1008af8ff0bd55a658affa9bea49d5481

Contents

`all_mailinglists` *Retrieve a data frame of all mailing lists from Qualtrics*

Description

Retrieve a data frame of all mailing lists from Qualtrics

Usage

```
all_mailinglists()
```

Details

If the request to the Qualtrics API made by this function fails, the request will be retried. If you see these failures on a 500 error (such as a 504 error) be patient while the request is retried; it will typically succeed on retrying. If you see other types of errors, retrying is unlikely to help.

Examples

```
## Not run:
# Register your Qualtrics credentials if you haven't already
qualtrics_api_credentials(
  api_key = "<YOUR-API-KEY>",
  base_url = "<YOUR-BASE-URL>"
)

# Retrieve a list of all mailing lists
mailinglists <- all_mailinglists()
```

```
## End(Not run)
```

`all_surveys` *Retrieve a data frame of all active surveys on Qualtrics*

Description

Retrieve a data frame of all active surveys on Qualtrics

Usage

```
all_surveys()
```

Details

If the request to the Qualtrics API made by this function fails, the request will be retried. If you see these failures on a 500 error (such as a 504 error) be patient while the request is retried; it will typically succeed on retrying. If you see other types of errors, retrying is unlikely to help.

See Also

See https://api.qualtrics.com/ for documentation on the Qualtrics API.

Examples

```
## Not run:
# Register your Qualtrics credentials if you haven't already
qualtrics_api_credentials(
  api_key = "<YOUR-API-KEY>",
  base_url = "<YOUR-BASE-URL>"
)

# Retrieve a list of all surveys
surveys <- all_surveys()

# Retrieve a single survey
mysurvey <- fetch_survey(surveyID = surveys$id[6])

mysurvey <- fetch_survey(
  surveyID = surveys$id[6],
  save_dir = tempdir(),
  start_date = "2018-01-01",
  end_date = "2018-01-31",
  limit = 100,
  label = TRUE,
  unanswer_recode = "UNANS",
  verbose = TRUE
)
```

```
## End(Not run)
```

`column_map` *Retrieve a data frame containing survey column mapping*

Description

Retrieve a data frame containing survey column mapping

Usage

```
column_map(surveyID)
```

Arguments

`surveyID`	A string. Unique ID for the survey you want to download. Returned as `id` by the all_surveys function.

Details

If the request to the Qualtrics API made by this function fails, the request will be retried. If you see these failures on a 500 error (such as a 504 error) be patient while the request is retried; it will typically succeed on retrying. If you see other types of errors, retrying is unlikely to help.

See Also

See https://api.qualtrics.com/ for documentation on the Qualtrics API.

Examples

```
## Not run:
# Register your Qualtrics credentials if you haven't already
qualtrics_api_credentials(
  api_key = "<YOUR-API-KEY>",
  base_url = "<YOUR-BASE-URL>"
)

# Retrieve a list of surveys
surveys <- all_surveys()

# Retrieve column mapping for a survey
mapping <- column_map(surveyID = surveys$id[6])

# Retrieve a single survey, filtering for specific questions
mysurvey <- fetch_survey(
  surveyID = surveys$id[6],
  save_dir = tempdir(),
  include_questions = c("QID1", "QID2", "QID3"),
```

```
  verbose = TRUE
)

## End(Not run)
```

`extract_colmap` *Extract column map from survey data download*

Description

Helper function to extract the column map attached to a response data download obtained from `fetch_survey()` (using the default `add_column_map = TRUE`)

Usage

```
extract_colmap(respdata)
```

Arguments

`respdata`	Response data including a column map dataframe as an attribute

Details

If the request to the Qualtrics API made by this function fails, the request will be retried. If you see these failures on a 500 error (such as a 504 error) be patient while the request is retried; it will typically succeed on retrying. If you see other types of errors, retrying is unlikely to help.

Examples

```
## Not run:
# Retrieve a list of surveys
surveys <- all_surveys()

# Retrieve a single survey
mysurvey <- fetch_survey(surveyID = surveys$id[6])

# Extract column mapping for survey
extract_colmap(mysurvey)

## End(Not run)
```

`fetch_description` *Download complete survey description using the Qualtrics v3 "Get Survey" API endpoint.*

Description

Download complete survey description using the Qualtrics v3 "Get Survey" API endpoint.

Usage

```
fetch_description(surveyID, elements = NULL, legacy = FALSE, ...)
```

Arguments

`surveyID`	A string. Unique ID for the survey you want to download. Returned as "id" by the all_surveys function.
`elements`	A character vector. Lists elements of survey definition to be maintained. Possible elements are "metadata", "surveyoptions", "flow", "blocks", "questions", "responsesets", and/or "scoring" (case-insensitive). If `legacy = TRUE`, then possible elements are "metadata", "questions", "responsecounts", "blocks", "flow", "embedded_data", and/or "comments".
`legacy`	Logical. If TRUE, will use older Get Survey API endpoint via a call to legacy function metadata.
`...`	Additional options, only used when `legacy = TRUE`. User may pass an argument called `questions`, a vector containing the names of questions for which you want to return metadata.

Details

If the request to the Qualtrics API made by this function fails, the request will be retried. If you see these failures on a 500 error (such as a 504 error) be patient while the request is retried; it will typically succeed on retrying. If you see other types of errors, retrying is unlikely to help.

Value

A list containing survey description metadata. The contents of the returned list depend on `elements`.

Examples

```
## Not run:
# Register your Qualtrics credentials if you haven't already
qualtrics_api_credentials(
  api_key = "<YOUR-API-KEY>",
  base_url = "<YOUR-BASE-URL>"
)

# Retrieve a list of surveys
```

```
surveys <- all_surveys()

# Get description for a survey
descrip <- fetch_description(surveyID = surveys$id[6])

# Get metadata with specific elements
descrip_specific <- fetch_description(
  surveyID = id,
  elements = c("questions", "flow")
)


## End(Not run)
```

`fetch_distributions` *Download distribution data for a survey from Qualtrics*

Description

Download distribution data for a survey from Qualtrics

Usage

```
fetch_distributions(surveyID)
```

Arguments

`surveyID`	String. Unique survey ID for the distribution data you want to download.

Details

If the request to the Qualtrics API made by this function fails, the request will be retried. If you see these failures on a 500 error (such as a 504 error) be patient while the request is retried; it will typically succeed on retrying. If you see other types of errors, retrying is unlikely to help.

Examples

```
## Not run:
# Register your Qualtrics credentials if you haven't already
qualtrics_api_credentials(
  api_key = "<YOUR-API-KEY>",
  base_url = "<YOUR-BASE-URL>"
)

surveys <- all_surveys()
distributions <- fetch_distributions(surveys$id[1])

## End(Not run)
```

fetch_distribution_history *Download distribution history data for a distribution from Qualtrics*

Description

Download distribution history data for a distribution from Qualtrics

Usage

```
fetch_distribution_history(distributionID)
```

Arguments

distributionID	String. Unique distribution ID for the distribution history you want to download.

Details

If the request to the Qualtrics API made by this function fails, the request will be retried. If you see these failures on a 500 error (such as a 504 error) be patient while the request is retried; it will typically succeed on retrying. If you see other types of errors, retrying is unlikely to help.

Examples

```
## Not run:
# Register your Qualtrics credentials if you haven't already
qualtrics_api_credentials(
  api_key = "<YOUR-API-KEY>",
  base_url = "<YOUR-BASE-URL>"
)

surveys <- all_surveys()
distributions <- fetch_distributions(surveys$id[1])
distribution_history <- fetch_distribution_history(distributions$id[1])

## End(Not run)
```

fetch_id *Fetch a unique Qualtrics survey ID based on survey name in the Qualtrics UI*

Description

Fetch a unique Qualtrics survey ID based on survey name in the Qualtrics UI

Usage

```
fetch_id(.data, survey_name, partial_match = FALSE)
```

Arguments

.data	Data frame of active surveys created by the function all_surveys().
survey_name	String. Name of the survey as it appears in the Qualtrics UI. Must be unique to be passed to fetch_id().
partial_match	Boolean. Will match all surveys containing the exact string provided. Defaults to FALSE, which matches against the entire name.

Details

Survey names in the Qualtrics platform are not required to be unique, but the survey_name argument for this function *must* be unique. If input results in multiple surveys being matched, will error with a list of up to 5 matches & their IDs

Examples

```
## Not run:
# Register your Qualtrics credentials if you haven't already
qualtrics_api_credentials(
  api_key = "<YOUR-API-KEY>",
  base_url = "<YOUR-BASE-URL>"
)

# Retrieve a list of surveys
surveys <- all_surveys()

# Retrieve surveyID for a unique survey
my_id <- fetch_id(surveys, "Unique Survey Name")


## End(Not run)
```

---

fetch_mailinglist *Download a mailing list from Qualtrics*

Description

Download a mailing list from Qualtrics

Usage

```
fetch_mailinglist(mailinglistID)
```

Arguments

`mailinglistID`	String. Unique ID for the mailing list you want to download. Returned as `id` by the all_mailinglists function.

Details

If the request to the Qualtrics API made by this function fails, the request will be retried. If you see these failures on a 500 error (such as a 504 error) be patient while the request is retried; it will typically succeed on retrying. If you see other types of errors, retrying is unlikely to help.

Examples

```
## Not run:
# Register your Qualtrics credentials if you haven't already
qualtrics_api_credentials(
  api_key = "<YOUR-API-KEY>",
  base_url = "<YOUR-BASE-URL>"
)

# Retrieve a list of all mailing lists
mailinglists <- all_mailinglists()

# Retrieve a single mailing list
mailinglist <- fetch_mailinglist(mailinglists$id[1])

## End(Not run)
```

`fetch_survey` *Download a survey and import it into R*

Description

Download a Qualtrics survey you own via API and import the survey directly into R.

Usage

```
fetch_survey(
  surveyID,
  limit = NULL,
  start_date = NULL,
  end_date = NULL,
  time_zone = NULL,
  include_display_order = TRUE,
  include_metadata = NULL,
  include_questions = NULL,
  include_embedded = NULL,
  unanswer_recode = NULL,
```

```
  unanswer_recode_multi = unanswer_recode,
  breakout_sets = TRUE,
  import_id = FALSE,
  label = TRUE,
  convert = TRUE,
  add_column_map = TRUE,
  add_var_labels = TRUE,
  strip_html = TRUE,
  col_types = NULL,
  verbose = TRUE,
  tmp_dir = tempdir(),
  last_response = deprecated(),
  force_request = deprecated(),
  save_dir = deprecated()
)
```

Arguments

`surveyID`	String. Unique ID for the survey you want to download. Returned as `id` by the all_surveys function.
`limit`	Integer. Maximum number of responses exported. Defaults to `NULL` (download all responses).
`start_date, end_date`	POSIXct, POSIXlt, or Date object, or length-1 string equivalent of form "YYYY-MM-DD" or "YYYY-MM-DD HH:MM:SS". ("/" is also acceptable in place of "-".) Only export survey responses that were **recorded** within the range specified by one or both arguments (i.e. referencing *RecordedDate*). Each defaults to `NULL` (unbounded). See Details for important information about both the package and Qualtrics' handling of start/end times.
`time_zone`	String. Time zone to use for date/time metadata variables in response dataframe (e.g. *StartDate*). Must match a time zone name from `base::OlsonNames()`. Defaults to `NULL`, which uses the current system timezone (from `base::Sys.timezone()`). Also applied to arguments `start_date` and/or `expiration_date` when given Date or string objects (see above); ignored when these arguments are given POSIXlt/POSIXct objects.
`include_display_order`	Logical. If `TRUE`, download from surveys using block/question/answer display randomization will include contain additional variables indicating the randomization pattern used for each case. Defaults to `FALSE`.
`include_metadata, include_questions, include_embedded`	Character vector. Specify variables to include in download. Defaults to `NULL` (keep all). `NA` or `character()` excludes all variables for that category. See Details for more on using each inclusion argument.
`unanswer_recode`	Integer-like. Recode seen-but-unanswered (usually skipped) questions using this value. Defaults to `NA`

`unanswer_recode_multi`
: Integer-like. Recode seen-but-unanswered multi-select questions (checkboxes) using this value. Defaults to value for `unaswer_recode`.

`breakout_sets`
: Logical. If `TRUE` multi-value fields (e.g. each option of a multi-select multiple choice questions) will be returned as separate columns. If `FALSE`, will be returned as 1 column with each element containing all values.

`import_id`
: Logical. If `TRUE`, column names will use Qualtrics import IDs (e.g. "QID123") instead of user-modifiable names (e.g. default names like "Q3" or custom names). Defaults to `FALSE` (user-modifiable names). Note that this also affects (otherwise unmodifiable) names of metadata columns–see the "`include_metadata`" section in Details below.

`label`
: Logical. If `TRUE` (default), will return text of answer choices, instead of recoded values (`FALSE`).

`convert`
: Logical. If `TRUE`, then the `fetch_survey()` function will convert certain question types (e.g. multiple choice) to proper data type in R. Defaults to `TRUE`.

`add_column_map`
: Logical. Add an attribute to the returned response data frame containing metadata associated with the response download, including variable names, question/choice text, and Qualtrics import IDs. This column map can be subsequently obtained using `extract_colmap()` Defaults to `TRUE`.

`add_var_labels`
: Logical. If `TRUE`, then the item description from each variable (equivalent to the one in the column map) will be added as a "label" attribute using `sjlabelled::set_label()`. Useful for reference as well as cross-compatibility with other stats packages (e.g., Stata, see documentation in `sjlabelled`). Defaults to `TRUE`.

`strip_html`
: Logical. If `TRUE`, then remove HTML tags from variable descriptions. Defaults to `TRUE`. Ignored if `add_column_map` and `add_var_labels` are both `FALSE`.

`col_types`
: Optional. This argument provides a way to manually overwrite column types that may be incorrectly guessed. Takes a `readr::cols()` specification. See example below and `readr::cols()` for formatting details. Defaults to `NULL`. Overwritten by `convert = TRUE`.

`verbose`
: Logical. If `TRUE`, verbose messages will be printed to the R console. Defaults to `TRUE`.

`tmp_dir`
: Path to filesystem directory. Qualtrics returns response data in compressed (zip) form. To extract raw data, the zip file must be briefly written to disk (the file is then promptly deleted). By default, the system's temporary directory is used for this (see `tempdir()`), but users needing more control can specify an alternate location here.

`last_response`
: Deprecated.

`force_request`
: Deprecated.

`save_dir`
: Deprecated.

Details

If the request to the Qualtrics API made by this function fails, the request will be retried. If you see these failures on a 500 error (such as a 504 error) be patient while the request is retried; it will typically succeed on retrying. If you see other types of errors, retrying is unlikely to help.

`start_date` & `end_date` arguments

The Qualtrics API endpoint for this function treats `start_date` and `end_date` slightly differently; `end_date` is *exclusive*, meaning only responses recorded up to the moment *before* the specified `end_date` will be returned. This permits easier automation; a previously-used `end_date` can become the `start_date` of a subsequent request without downloading duplicate records.

As a convenience for users working interactively, the qualtRics package also accepts Date(-like) input to each argument, which when used implies a time of 00:00:00 on the given date (and time zone). When a Date(-like) is passed to `end_date`, however, the date will be incremented by one before making the API request. This adjustment is intended to provide interactive users with more intuitive results; for example, specifying "2022/06/02" for both `start_date` and `end_date` will return all responses for that day, (instead of the zero responses that would return if `end_date` was not adjusted).

Inclusion/exclusion arguments

The three `include_*` arguments each have different requirements:

`include_metadata`**:**

Elements must be one of the 17 Qualtrics metadata variables, listed here in their default order: *StartDate* (*startDate*), *EndDate* (*endDate*), *Status* (*status*), *IPAddress* (*ipAddress*), *Progress* (*progress*), *Duration (in seconds)* (*duration*), *Finished* (*finished*), *RecordedDate* (*recordedDate*), *ResponseId* (*_recordId*), *RecipientLastName* (*recipientLastName*), *RecipientFirstName* (*recipientFirstName*), *RecipientEmail* (*recipientEmail*), *ExternalReference* (*externalDataReference*), *LocationLatitude* (*locationLatitude*), *LocationLongitude* (*locationLongitude*), *DistributionChannel* (*distributionChannel*), *UserLanguage* (*userLanguage*).

Names in parentheses are those returned by the API endpoint when `import_id` is set to `TRUE`. The argument `include_metadata` can accept either format regardless of `import_id` setting, and names are not case-sensitive. Duplicate elements passed to `include_metadata` will be silently dropped, with the de-duplicated variable located in the first position.

`include_questions`**:**

Qualtrics uniquely identifies each question with an internal ID that takes the form "QID" followed by a number, e.g. *QID5*. When using `include_questions`, these internal IDs must be used rather than user-customizable variable names (which need not be unique in Qualtrics). If needed, a column map linking customizable names to QID's can be quickly obtained by calling:

```
my_survey <- fetch_survey(
    surveyID = {survey ID},
    limit = 1,
    add_column_map = TRUE
)
extract_colmap(my_survey)
```

Note that while there is one QID for each "question" in the Qualtrics sense, each QID may still map to multiple columns in the returned data frame. If, for example, a "question" with ID *QID5* is a multiple-choice item with a text box added to the third choice, the returned data frame may have two related columns: *"QID5"* for the multiple choice selection, and *"QID5_3_TEXT"* for the text box (or, more typically, their custom names). Setting `include_questions = "QID5"` will always return both columns. Similarly, "matrix" style multiple-choice questions will have a column for

each separate row of the matrix. Also, when `include_display_order = TRUE`, display ordering variables for any randomization will be included. Currently, separating these sub-questions via the API does not appear possible (e.g., `include_questions = "QID5_3_TEXT"` will result in an API error).

`include_embedded`**:**

This argument accepts the user-specified names of any embedded data variables in the survey being accessed.

See Also

See https://api.qualtrics.com/ for documentation on the Qualtrics API.

Examples

```
## Not run:
# Register your Qualtrics credentials if you haven't already
qualtrics_api_credentials(
  api_key = "<YOUR-API-KEY>",
  base_url = "<YOUR-BASE-URL>"
)

# Retrieve a list of surveys
surveys <- all_surveys()

# Retrieve a single survey
my_survey <- fetch_survey(surveyID = surveys$id[6])

my_survey <- fetch_survey(
  surveyID = surveys$id[6],
  start_date = "2018-01-01",
  end_date = "2018-01-31",
  limit = 100,
  label = TRUE,
  unanswer_recode = 999,
  verbose = TRUE,
  # Manually override EndDate to be a character vector
  col_types = readr::cols(EndDate = readr::col_character())
)


## End(Not run)
```

`list_distribution_links` *Download distribution links for a distribution from Qualtrics*

Description

Download distribution links for a distribution from Qualtrics

Usage

```
list_distribution_links(distributionID, surveyID)
```

Arguments

distributionID	String. Unique distribution ID for the distribution links you want to download.
surveyID	String. Unique ID for the survey you want to download.

Details

If the request to the Qualtrics API made by this function fails, the request will be retried. If you see these failures on a 500 error (such as a 504 error) be patient while the request is retried; it will typically succeed on retrying. If you see other types of errors, retrying is unlikely to help.

Examples

```
## Not run:
# Register your Qualtrics credentials if you haven't already
qualtrics_api_credentials(
  api_key = "<YOUR-API-KEY>",
  base_url = "<YOUR-BASE-URL>"
)

surveys <- all_surveys()
distributions <- fetch_distributions(surveys$id[1])
distribution_links <- list_distribution_links(distributions$id[1], surveyID = surveys$id[1])

## End(Not run)
```

---

metadata *Download metadata for a survey*

Description

Using this function, you can retrieve metadata about your survey. This information includes question metadata (type, options, choices, etc), number of responses, general metadata, survey flow, etc.

Usage

```
metadata(surveyID, get = NULL, questions = NULL)
```

Arguments

surveyID	A string. Unique ID for the survey you want to download. Returned as "id" by the all_surveys function.
get	A character vector containing any of the following: "metadata", "questions", "responsecounts", "blocks", "flow", "embedded_data", or "comments". Will return included elements. By default, the function returns the "metadata", "questions", and "responsecounts" elements. See examples below for more information.
questions	Character vector containing the names of questions for which you want to return metadata. Defaults to NULL (all questions).

Details

If the request to the Qualtrics API made by this function fails, the request will be retried. If you see these failures on a 500 error (such as a 504 error) be patient while the request is retried; it will typically succeed on retrying. If you see other types of errors, retrying is unlikely to help.

Examples

```
## Not run:
# Register your Qualtrics credentials if you haven't already
qualtrics_api_credentials(
  api_key = "<YOUR-API-KEY>",
  base_url = "<YOUR-BASE-URL>"
)

# Retrieve a list of surveys
surveys <- all_surveys()

# Get metadata for a survey
md <- metadata(surveyID = surveys$id[6])

# Get metadata with specific elements
md_specific <- metadata(
  surveyID = id,
  get = c("flow")
)

# Get specific question metadata
question_specific <- metadata(
  surveyID = id,
  get = c("questions"),
  questions = c("Q1", "Q2")
)

# Example of a metadata file
file <- system.file("extdata", "metadata.rds", package = "qualtRics")

# Load
metadata_ex <- readRDS(file = file)

## End(Not run)
```

qualtrics_api_credentials *Install Qualtrics credentials in your* .Renviron *file for repeated use*

Description

This function adds your Qualtrics API key and base URL to your .Renviron file so it can be called securely without being stored in your code. After you have installed these two credentials, they can be called any time with Sys.getenv("QUALTRICS_API_KEY") or Sys.getenv("QUALTRICS_BASE_URL"). If you do not have an .Renviron file, the function will create one for you. If you already have an .Renviron file, the function will append the key to your existing file, while making a backup of your original file for disaster recovery purposes.

Usage

```
qualtrics_api_credentials(
  api_key,
  base_url,
  overwrite = FALSE,
  install = FALSE,
  report = FALSE
)
```

Arguments

api_key	The API key provided to you from Qualtrics formatted in quotes. Learn more about Qualtrics API keys at https://api.qualtrics.com/
base_url	The institution-specific base URL for your Qualtrics account, formatted in quotes, without the protocol (do not include https://). Find your base URL at https://api.qualtrics.com/
overwrite	If TRUE, will overwrite existing Qualtrics credentials that you already have in your .Renviron file.
install	If TRUE, will install the key in your .Renviron file for use in future sessions. Defaults to FALSE (single session use).
report	If TRUE, ignores other arguments and outputs credentials as a 2-element named vector.

Examples

```
## Not run:
qualtrics_api_credentials(
  api_key = "<YOUR-QUALTRICS_API_KEY>",
  base_url = "<YOUR-QUALTRICS_BASE_URL>",
  install = TRUE
)
```

```
# Reload your environment so you can use the credentials without restarting R
readRenviron("~/.Renviron")
# You can check it with:
Sys.getenv("QUALTRICS_API_KEY")

# If you need to overwrite existing credentials:
qualtrics_api_credentials(
  api_key = "<YOUR-QUALTRICS_API_KEY>",
  base_url = "<YOUR-QUALTRICS_BASE_URL>",
  overwrite = TRUE,
  install = TRUE
)
# Reload your environment to use the credentials

## End(Not run)
```

`read_survey` *Read a CSV file exported from Qualtrics*

Description

Reads comma separated CSV files generated by Qualtrics software. The second line containing the variable labels is imported. Repetitive introductions to matrix questions are automatically removed. Variable labels are stored as attributes.

Usage

```
read_survey(
  file_name,
  strip_html = TRUE,
  import_id = FALSE,
  time_zone = NULL,
  legacy = FALSE,
  add_column_map = TRUE,
  add_var_labels = TRUE,
  col_types = NULL
)
```

Arguments

`file_name`	String. A CSV data file.
`strip_html`	Logical. If `TRUE`, then remove HTML tags from variable descriptions. Defaults to `TRUE`.
`import_id`	Logical. If `TRUE`, use Qualtrics import IDs instead of question IDs as column names. Defaults to `FALSE`.
`time_zone`	String. A local timezone to determine response date values. Defaults to `NULL` which corresponds to UTC time. See "Dates and Times" from Qualtrics for more information on format.

legacy	Logical. If TRUE, then import "legacy" format CSV files (as of 2017). Defaults to FALSE.
add_column_map	Logical. If TRUE, then a column map data frame will be added as an attribute to the main response data frame. This column map captures Qualtrics-provided metadata associated with the response download, such as an item description and internal ID's. Defaults to TRUE.
add_var_labels	Logical. If TRUE, then the item description from each variable (equivalent to the one in the column map) will be added as a "label" attribute using sjlabelled::set_label(). Useful for reference as well as cross-compatibility with other stats packages (e.g., Stata, see documentation in sjlabelled). Defaults to TRUE.
col_types	Optional. This argument provides a way to manually overwrite column types that may be incorrectly guessed. Takes a readr::cols() specification. See example below and readr::cols() for formatting details. Defaults to NULL.

Value

A data frame. Variable labels are stored as attributes. They are not printed on the console but are visibile in the RStudio viewer.

Examples

```
## Not run:
# Generic use of read_survey()
df <- read_survey("<YOUR-PATH-TO-CSV-FILE>")

## End(Not run)
# Example using current data format
file <- system.file("extdata", "sample.csv", package = "qualtRics")
df <- read_survey(file)

# Example using legacy data format
file <- system.file("extdata", "sample_legacy.csv", package = "qualtRics")
df <- read_survey(file, legacy = TRUE)

# Example changing column type
file <- system.file("extdata", "sample.csv", package = "qualtRics")
# Force EndDate to be a string
df <- read_survey(file, col_types = readr::cols(EndDate = readr::col_character()))
```

---

survey_questions *Retrieve a data frame containing question IDs and labels*

Description

Retrieve a data frame containing question IDs and labels

Usage

```
survey_questions(surveyID)
```

Arguments

surveyID	A string. Unique ID for the survey you want to download. Returned as `id` by the all_surveys function.

Details

If the request to the Qualtrics API made by this function fails, the request will be retried. If you see these failures on a 500 error (such as a 504 error) be patient while the request is retried; it will typically succeed on retrying. If you see other types of errors, retrying is unlikely to help.

See Also

See https://api.qualtrics.com/ for documentation on the Qualtrics API.

Examples

```
## Not run:
# Register your Qualtrics credentials if you haven't already
qualtrics_api_credentials(
  api_key = "<YOUR-API-KEY>",
  base_url = "<YOUR-BASE-URL>"
)

# Retrieve a list of surveys
surveys <- all_surveys()

# Retrieve questions for a survey
questions <- survey_questions(surveyID = surveys$id[6])

# Retrieve a single survey, filtering for specific questions
mysurvey <- fetch_survey(
  surveyID = surveys$id[6],
  save_dir = tempdir(),
  include_questions = c("QID1", "QID2", "QID3"),
  verbose = TRUE
)

## End(Not run)
```

Index