

Package: robotoolbox (via r-universe)

October 10, 2026

Title Client for the 'KoboToolbox' API

Version 1.6.2

Description Suite of utilities for accessing and manipulating data from the 'KoboToolbox' API. 'KoboToolbox' is a robust platform designed for field data collection in various disciplines. This package aims to simplify the process of fetching and handling data from the API. Detailed documentation for the 'KoboToolbox' API can be found at <<https://support.kobotoolbox.org/api.html>>.

Depends R (>= 4.1)

License MIT + file LICENSE

Encoding UTF-8

URL <https://dickoa.gitlab.io/robotoolbox>,
<https://gitlab.com/dickoa/robotoolbox>

BugReports <https://gitlab.com/dickoa/robotoolbox/-/issues>

Imports crul (>= 1.4.0), RcppSimdJson (>= 0.1.6), data.table (>= 1.14.2), dplyr (>= 1.1.2), tidyr (>= 1.3.0), purrr (>= 1.0.1), rlang (>= 1.0.0), tidyselect (>= 1.2.0), tibble (>= 3.2.1), stringi (>= 1.7.6), glue (>= 1.6.0), dm (>= 1.0.10), labelled (>= 2.11.0), readr (>= 2.1.0), cli (>= 3.6.1)

Suggests roxygen2 (>= 7.2.3), devtools (>= 2.4.3), vcr (>= 1.2.0), knitr (>= 1.37), testthat (>= 3.1.1), covr (>= 3.6.2), rmarkdown (>= 2.21), DiagrammeR (>= 1.0.9), DiagrammeRsvg (>= 0.1), sf (>= 1.0.9), mapview (>= 2.11.0)

VignetteBuilder knitr

RoxygenNote 7.3.3

Roxygen list(markdown = TRUE)

LazyData true

Config/Needs/website unhcr-dataviz/unhcrtemplate

X-schema.org-applicationCategory Data Access

X-schema.org-keywords open-data, kobotoolbox, odk, kpi, api, data, dataset

Config/pak/sysreqs make libicu-dev libssl-dev libx11-dev zlib1g-dev
Repository https://humanitarian-user-group.r-universe.dev
Date/Publication 2026-04-05 13:26:54 UTC
RemoteUrl https://github.com/dickoa/robotoolbox
RemoteRef HEAD
RemoteSha a27a76621507f0fbecac59f78faa8baec89d6160

Contents

asset_list	2
data_ml_en	3
kobo_asset	4
kobo_asset_file_list	5
kobo_asset_list	6
kobo_asset_version	7
kobo_asset_version_list	7
kobo_attachment_download	8
kobo_audit	9
kobo_cache	10
kobo_cache_clear	11
kobo_cache_info	11
kobo_data	12
kobo_form	15
kobo_lang	16
kobo_settings	18
kobo_setup	19
kobo_token	20
Index	21

asset_list	<i>Examples of KoboToolbox assets and list of assets</i>
------------	--

Description

Examples of KoboToolbox assets and list of assets.

Usage

asset_list

asset_ml

asset_rg

asset_spatial

asset_sm_label

asset_audit

Format

asset_list: a `data.frame` of 28 rows and 7 columns with a list of API assets

asset_ml: A `kobo_asset` object on a survey using multiple languages.

asset_rg: A `kobo_asset` object on a survey using repeat groups.

asset_spatial: A `kobo_asset` object on a survey showcasing gps questions.

asset_sm_label: A `kobo_asset` object to showcase select multiple labels.

asset_audit: A `kobo_asset` object on a survey with audit logging enabled.

data_ml_en

Examples of KoboToolbox submissions data

Description

Examples of KoboToolbox submissions data.

Usage

data_ml_en

data_ml_fr

data_ml_ar

data_ml_default

data_ml_vlabel

data_rg

data_spatial

data_sm

data_sm_label

data_audit

Format

data_ml: A data.frame with submissions from [asset_ml](#) in English.

data_ml_fr: A data.frame with submissions from [asset_ml](#) in French.

data_ml_ar: A data.frame with submissions from [asset_ml](#) in Arabic

data_ml_default: A data.frame with submissions from [asset_ml](#) with the default language.

data_ml_vlabel: A data.frame with submissions from [asset_ml](#) using variable labels as column names.

data_rg: A dm object with submissions from [asset_rg](#)

data_spatial: A data.frame with submissions from the [asset_spatial](#) KoboToolbox API asset.

data_sm: A data.frame with submissions from [asset_sm_label](#) with no labels for the select_multiple question.

data_sm_label: A data.frame with submissions from [asset_sm_label](#) with labels for the select_multiple question.

data_audit: A data.frame with submissions from [asset_audit](#).

kobo_asset

Get a specific KoboToolbox API asset from a unique identifier

Description

Get a specific KoboToolbox API asset from a unique identifier

Usage

```
kobo_asset(x)
```

Arguments

x the unique identifier of a specific asset (character) or a kobo_asset object.

Value

A kobo_asset object. It contains all the information about the KoboToolbox API asset associated to the unique identifier.

Examples

```
## Not run:
# replace by your own url and token
kobo_setup(url = "https://kf.kobotoolbox.org", token = "abcde")
# use a valid uid
uid <- "a9cwEQcbWqWzA5hzkjRUWi"
asset <- kobo_asset(uid)
asset

## End(Not run)
```

kobo_asset_file_list *List all uploaded files related to a KoboToolbox API asset*

Description

List all uploaded files related to a KoboToolbox API asset

Usage

```
kobo_asset_file_list(x)
```

Arguments

x the asset uid or the kobo_asset object.

Value

A data.frame containing the list of all your KoboToolbox API files under the asset:

- uid the asset unique identifier
- url url of the files API endpoint
- asset url of the files associated asset API endpoint
- user the user account of the owner of the asset
- user__username when the asset was created
- file_type files type either form_media or map_layer
- description files description
- date_created date when the files were created
- content url to download the files
- hashmd5 hash of the files
- filename names of the files
- mimetype mime type of the files

Examples

```
## Not run:  
kobo_setup()  
uid <- "a9cwEQcbWqWzA5hzkjRUWi"  
kobo_file_list(uid)  
  
## End(Not run)
```

kobo_asset_list	<i>List all available KoboToolbox API assets</i>
-----------------	--

Description

List all available KoboToolbox API assets and their metadata.

Usage

```
kobo_asset_list(limit = 100L)
```

Arguments

`limit` integer, the number of API assets to display per page. Default to 100.

Value

A data.frame containing the list of all your KoboToolbox API assets and the following metadata:

- `uid` the asset unique identifier
- `name` the name of the asset
- `asset_type` the type of asset (block, survey, question, or template)
- `owner_username` the user account of the owner of the asset
- `date_create` when the asset was created
- `date_modified` when the asset was last modified
- `deployed` whether or not the asset is currently deployed
- `submissions` the number of submissions for the asset (survey)

Examples

```
## Not run:  
kobo_setup()  
asset_list <- kobo_asset_list(limit = 10L)  
asset_list  
  
## End(Not run)
```

kobo_asset_version	<i>Get a specific KoboToolbox API asset version from an asset unique identifier</i>
--------------------	---

Description

Get a specific KoboToolbox Asset version from an asset unique identifier or kobo_asset object

Usage

```
kobo_asset_version(x, version)
```

Arguments

x	the unique identifier of a specific asset (character) or a kobo_asset object.
version	character, the unique identifier of the version of the asset

Value

A kobo_asset_version object

Examples

```
## Not run:
kobo_setup()
uid <- "a9cwEQcbWqWzA5hzkjRUWi"
asset <- kobo_asset(uid)
asset_version_list <- kobo_asset_version_list(asset)
kobo_asset_version(asset, asset_version_list$uid[1])

## End(Not run)
```

kobo_asset_version_list	<i>List all available versions of a KoboToolbox API asset</i>
-------------------------	---

Description

List all available versions of a KoboToolbox API asset and their metadata.

Usage

```
kobo_asset_version_list(x)
```

Arguments

x the uid or kobo_asset object.

Value

A data.frame containing the list of all the versions of a given KoboToolbox API asset with the following metadata:

- uid the asset version unique identifier.
- url the URL of the asset version.
- deployed whether or not the asset version is deployed
- date_modified when the asset version was last modified

a data.frame

Examples

```
## Not run:
kobo_setup() # setup using your url and token
uid <- "a9cwEQcbWqWzA5hzkjRUWi" # pick a valid uid
asset <- kobo_asset(uid)
kobo_asset_version_list(asset)

## End(Not run)
```

kobo_attachment_download

Download submitted files associated to KoboToolbox API asset

Description

Download submitted files associated to a KoboToolbox API asset

Usage

```
kobo_attachment_download(x, folder, progress, overwrite, n_retry)
```

Arguments

x	the asset uid or the kobo_asset object.
folder	character, the folder where you store the downloaded files.
progress	logical, whether or not you want to see the progress via message. Default to FALSE.
overwrite	logical, whether or not you want to overwrite existing media files. Default to TRUE.
n_retry	integer, Number of time you should retry the failed request. Default to 3L.

Value

Silently returns a vector of files paths.

Examples

```
## Not run:
kobo_setup()
uid <- "a9cwEQcbWqWzA5hzkjRUWi"
kobo_attachment_download(uid, folder = tempdir())

## End(Not run)
```

kobo_audit

Get all audit logs data from a KoboToolbox survey

Description

Get all audit logs data from a KoboToolbox survey through a kobo_asset or asset unique identifier.

Usage

```
kobo_audit(x, progress)
```

Arguments

x	the unique identifier of a specific asset (character) or a kobo_asset object.
progress	logical, whether or not you want to see the progress via message. Default to FALSE.

Value

A data.frame. It contains survey paradata from audit logs. The following columns are available:

- `_id` This columns generated by robotoolbox allow you to do a mapping the `_id` of the submissions in `kobo_data`.
- `event` the action that took place. The different event types include. form start, form exit, question, group questions, end screen, and device or metadata audit.
- `node` the name of the question or group related to the event.
- `name` This column is appended by robotoolbox to match the name of the question in the audit and the data from `kobo_data`.
- `start` the timestamp when the event started.
- `end` the timestamp when the event ended.
- `latitude` the latitude of the device when the event occurred.
- `longitude` the longitude of the device when the event occurred.

- accuracy the GPS accuracy of the location data.
- old-value the previous value of the question before it was changed in this event.
- new-value the new value of the question after it was changed in this event.
- user the username of the data collector.
- change-reason the reason before they save changes to a form.

Examples

```
## Not run:
kobo_setup()
uid <- "a9cwEQcbWqWzdA5eqkjRUWi"
asset <- kobo_asset(uid)
audit <- kobo_audit(asset)

if (require(dplyr)) {
  library(dplyr)
  glimpse(audit)
}

## End(Not run)
```

kobo_cache

Session-level cache for KoboToolbox metadata

Description

Session-level caching for form metadata and languages. Form versions are immutable and languages don't change, so caching is safe and significantly improves performance.

Details

Cached items:

- **Form metadata:** When `all_versions = TRUE` in `kobo_data()`, multiple API calls are made to fetch form schemas for each version. Caching eliminates redundant calls within a session.
- **Languages:** `kobo_lang()` results are cached per asset, speeding up `kobo_lang_set()` and `kobo_lang_get()` operations.

Cache is stored in memory and cleared when R session ends.

Note: Version lists are NOT cached as they can change when users deploy new form versions, and the API call sequence is required by the server.

kobo_cache_clear	<i>Clear the metadata cache</i>
------------------	---------------------------------

Description

Clear the metadata cache

Usage

```
kobo_cache_clear(uid = NULL, type = c("all", "forms", "langs"))
```

Arguments

uid	Optional asset uid. If provided, clears cache for that asset only.
type	Character, which cache to clear: "all" (default), "forms", or "langs".

Value

Invisibly returns TRUE

Examples

```
## Not run:  
kobo_cache_clear()  
kobo_cache_clear("aXf4gH7kL9mN2pQrStUvWx")  
kobo_cache_clear(type = "langs")  
  
## End(Not run)
```

kobo_cache_info	<i>Get cache statistics</i>
-----------------	-----------------------------

Description

Get cache statistics

Usage

```
kobo_cache_info()
```

Value

A list with cache statistics

Examples

```
## Not run:
kobo_cache_info()

## End(Not run)
```

kobo_data*Get all submissions from a KoboToolbox API asset*

Description

Get all submissions from a KoboToolbox API asset through a kobo_asset or asset unique identifier.

Usage

```
kobo_data(
  x,
  lang,
  all_versions,
  colnames_label,
  select_multiple_label,
  select_multiple_sep,
  progress,
  paginate,
  page_size,
  query,
  fields
)

kobo_submissions(
  x,
  lang,
  all_versions,
  colnames_label,
  select_multiple_label,
  select_multiple_sep,
  progress,
  paginate,
  page_size,
  query,
  fields
)

## S3 method for class 'kobo_asset'
kobo_submissions(
  x,
```

```

    lang = NULL,
    all_versions = TRUE,
    colnames_label = FALSE,
    select_multiple_label = FALSE,
    select_multiple_sep = "_",
    progress = FALSE,
    paginate = NULL,
    page_size = NULL,
    query = NULL,
    fields = NULL
)

## S3 method for class 'character'
kobo_submissions(
  x,
  lang = NULL,
  all_versions = TRUE,
  colnames_label = FALSE,
  select_multiple_label = FALSE,
  select_multiple_sep = "_",
  progress = FALSE,
  paginate = NULL,
  page_size = NULL,
  query = NULL,
  fields = NULL
)

## Default S3 method:
kobo_submissions(
  x,
  lang = NULL,
  all_versions = TRUE,
  colnames_label = FALSE,
  select_multiple_label = FALSE,
  select_multiple_sep = "_",
  progress = FALSE,
  paginate = NULL,
  page_size = NULL,
  query = NULL,
  fields = NULL
)

```

Arguments

<code>x</code>	the asset uid or the kobo_asset object.
<code>lang</code>	character, form language used for the variable and value labels.
<code>all_versions</code>	logical, whether or not to include submissions from all form versions. Default to TRUE. If FALSE, it uses the data from the latest version of the form.

colnames_label	logical, whether or not to use variable labels in lieu of column names based on form question names. Default to FALSE.
select_multiple_label	logical, whether or not to replace select_multiple columns values by labels. Default to FALSE.
select_multiple_sep	character, column and choices separator for newly created dummy variables. Default to "_".
progress	logical, whether or not you want to see the progress via message. Default to FALSE.
paginate	logical, split submissions by page_size. Default to NULL.
page_size	integer, number of submissions per page.
query	character, a MongoDB-style query string to filter submissions server-side. For example, '{"field_name": "value"}' to return only submissions where field_name equals "value". The query filters submissions (main table rows) based on top-level fields only. You cannot filter on fields inside a repeat group. All repeat group rows belonging to matching submissions are returned. Default to NULL (no filtering).
fields	character vector of field names to return. When provided, only these fields (plus __version__) are fetched from the server. The server may still include system fields such as _id and _uuid. For repeat-group forms, fields supports individual selection of main-table columns and table-level selection of top-level repeat groups. A selected repeat group is always returned in full. Selecting individual child columns within a repeat group (for example, "my_repeat/age") is not supported. Nested repeat groups cannot be selected independently, you'll need to include their ancestor repeat group instead. Repeat-group forms always return a dm object. If no repeat group is selected, the returned dm contains only the main table. Default to NULL (all fields).

Details

[kobo_data](#) is the main function of robotoolbox, it is used pull submissions from your Kobotoolbox survey. The main result is a data.frame for regular form and you have a dm for a form with repeating groups of questions.

Value

A data.frame or A dm object if you have a repeating group of questions. It contains the responses from the Kobotoolbox survey.

Examples

```
## Not run:
# Use your own URL and token
kobo_setup(url = "https://kf.kobotoolbox.org/",
            token = "9et1814c285w094f6v9bd629df47a1a0e81x53a0")
# Use your own unique identifier
```

```

uid <- "a9cwEQcbWqWzA5hzkjRUWi"
asset <- kobo_asset(uid)
subs <- kobo_data(asset)

# Filter submissions server-side with a MongoDB query
filtered <- kobo_data(asset, query = '{"pet_yesno": "1"}')

# Select specific fields
partial <- kobo_data(asset, fields = c("full_name", "pet_yesno"))

if (require(dplyr)) {
  library(dplyr)
  glimpse(subs)
}

## End(Not run)

```

kobo_form

Get a KoboToolbox survey form

Description

Get a KoboToolbox survey form from a kobo_asset or an asset unique identifier.

Usage

```
kobo_form(x, version)
```

Arguments

x	the unique identifier of a specific asset (character) or a kobo_asset object.
version	character, the unique identifier of the version of the asset.

Value

A data.frame with the following columns:

- name the name of the survey questions
- list_name the name of list of code used for values and labels
- type the type of KoboToolbox survey questions
- label the label of the questions
- lang the languages used in the survey
- version the survey version unique identifier
- choices a list column with the choices values and labels
- kuid the unique identifier of the question
- qpath and xpath the path of the question in JSON/XML

You can also have other columns such as relevant, calculation, etc. depending on how you structure for survey form.

Examples

```
## Not run:
# Use your own URL and token
kobo_setup(url = "https://kf.kobotoolbox.org/",
           token = "9et1814c285w094f6v9bd629df47a1a0e81x53a0")
# Use your own API asset identifier
uid <- "a9cwEQcbWqWzA5hzkjRUWi"
asset <- kobo_asset(uid)
form <- kobo_form(asset)

## End(Not run)
```

kobo_lang

Get and set the languages used in KoboToolbox data

Description

- `kobo_lang()`: Get the languages available in a KoboToolbox survey form.
- `kobo_lang_get()`: Get the current language of a dataset returned by `kobo_data()`.
- `kobo_lang_set()`: Change the language labels of a dataset without re-downloading.

Usage

```
kobo_lang(x)

## S3 method for class 'kobo_asset'
kobo_lang(x)

## S3 method for class 'character'
kobo_lang(x)

## Default S3 method:
kobo_lang(x)

kobo_lang_get(data, asset)

## S3 method for class 'data.frame'
kobo_lang_get(data, asset)

## S3 method for class 'dm'
kobo_lang_get(data, asset)

## Default S3 method:
kobo_lang_get(data, asset)

kobo_lang_set(data, asset, lang)
```

```
## S3 method for class 'data.frame'
kobo_lang_set(data, asset, lang)

## S3 method for class 'dm'
kobo_lang_set(data, asset, lang)

## Default S3 method:
kobo_lang_set(data, asset, lang)
```

Arguments

x	For kobo_lang(): the unique identifier of a specific asset (character) or a kobo_asset object.
data	For kobo_lang_get() and kobo_lang_set(): a data.frame or dm object returned by kobo_data().
asset	For kobo_lang_get() and kobo_lang_set(): a kobo_asset object or the unique identifier of a specific asset (character).
lang	For kobo_lang_set(): character, the target language. Use kobo_lang(asset) to see available languages.

Details

KoboToolbox forms can have multiple language translations. Normally, switching languages requires calling kobo_data() again with a different lang parameter, which re-downloads all data from the server.

kobo_lang_set() applies labels from a different language to existing data instantly, using the cached form metadata. This is useful when you need to work with data in multiple languages or export data with different label sets.

kobo_lang_get() inspects the current variable labels in the data and matches them against the form to determine which language is currently applied.

Value

- kobo_lang(): A character vector of available languages. Returns "Labels" when no language is set.
- kobo_lang_get(): A character string of the current language, or NA if it cannot be determined.
- kobo_lang_set(): The data with updated labels in the new language.

Examples

```
## Not run:
# Setup
asset <- kobo_asset("aYuTZn9vegi3Z49MXwKjep")

# List available languages
kobo_lang(asset)
```

```

# Fetch data in English
df <- kobo_data(asset, lang = "English (en)")

# Check current language (using asset object or uid)
kobo_lang_get(df, asset)
kobo_lang_get(df, "aYuTZn9vegi3Z49MXwKjep")

# Switch to French instantly (no API call)
df_fr <- kobo_lang_set(df, asset, lang = "Francais (fr)")
kobo_lang_get(df_fr, asset)

# Works with dm objects (nested forms) too
dm_data <- kobo_data(complex_asset)
dm_fr <- kobo_lang_set(dm_data, complex_asset, lang = "French")

## End(Not run)

```

kobo_settings	<i>Get robotoolbox settings</i>
---------------	---------------------------------

Description

Print the KoboToolbox server URL and API token currently in use.

Usage

```
kobo_settings()
```

Value

A list with information about your KoboToolbox server URL and token.

Examples

```

## Not run:
kobo_settings()

## End(Not run)

```

kobo_setup	<i>Set robotoolbox settings</i>
------------	---------------------------------

Description

Set the KoboToolbox server URL, API token and return invisibly a kobo_settings object.

Usage

```
kobo_setup(
    url = Sys.getenv("KOBOTOOLBOX_URL", ""),
    token = Sys.getenv("KOBOTOOLBOX_TOKEN", ""),
    page_size = 1000L
)
```

Arguments

url	character, the base URL of the KoboToolbox server.
token	character, the API token.
page_size	integer, the maximum number of submissions per API request. Default to 1000L, which is the limit enforced by public KoboToolbox servers since KoboToolbox KPI version 2.026.03 (March 2026). Users with private servers that allow higher limits can set this to a larger value (e.g., 30000L) to fetch more data per request and reduce the number of paginated calls.

Value

A kobo_settings object printing the server URL and the API token.

Examples

```
## Not run:
# Public server (default page_size = 1000)
kobo_setup(url = "https://kf.kobotoolbox.org/",
            token = "9et1814c285w094f6v9bd629df47a1a0e81x53a0")

# Private server with higher limit
kobo_setup(url = "https://myserver.org/",
            token = "9et1814c285w094f6v9bd629df47a1a0e81x53a0",
            page_size = 30000)
kobo_settings()

## End(Not run)
```

kobo_token	<i>Get your KoboToolbox API token</i>
------------	---------------------------------------

Description

Get your KoboToolbox API token from your username and password.

Usage

```
kobo_token(username = NULL, password = NULL, url = NULL, overwrite = FALSE)
```

Arguments

username	character, KoboToolbox account username.
password	character, KoboToolbox account password.
url	character, KoboToolbox server URL.
overwrite	logical, if TRUE, it overwrites the existing token. Default to FALSE.

Value

A character, the KoboToolbox API token. It also stores, as a side effect, the URL and token as the environment variables KOBOTOOLBOX_URL and KOBOTOOLBOX_TOKEN respectively.

Examples

```
## Not run:
# use your own KoboToolbox URL, username and password
if (require(askpass)) {
  token <- kobo_token(username = "cool_user_name",
                      password = askpass::askpass(),
                      url = "https://kf.kobotoolbox.org/")

  token
}

## End(Not run)
```

Index

* datasets

- asset_list, [2](#)
- data_ml_en, [3](#)
- asset_audit, [4](#)
- asset_audit (asset_list), [2](#)
- asset_list, [2](#)
- asset_ml, [4](#)
- asset_ml (asset_list), [2](#)
- asset_rg, [4](#)
- asset_rg (asset_list), [2](#)
- asset_sm_label, [4](#)
- asset_sm_label (asset_list), [2](#)
- asset_spatial, [4](#)
- asset_spatial (asset_list), [2](#)
- data_audit (data_ml_en), [3](#)
- data_ml_ar (data_ml_en), [3](#)
- data_ml_default (data_ml_en), [3](#)
- data_ml_en, [3](#)
- data_ml_fr (data_ml_en), [3](#)
- data_ml_vlabel (data_ml_en), [3](#)
- data_rg (data_ml_en), [3](#)
- data_sm (data_ml_en), [3](#)
- data_sm_label (data_ml_en), [3](#)
- data_spatial (data_ml_en), [3](#)
- kobo_asset, [4](#)
- kobo_asset_file_list, [5](#)
- kobo_asset_list, [6](#)
- kobo_asset_version, [7](#)
- kobo_asset_version_list, [7](#)
- kobo_attachment_download, [8](#)
- kobo_audit, [9](#)
- kobo_cache, [10](#)
- kobo_cache_clear, [11](#)
- kobo_cache_info, [11](#)
- kobo_data, [12](#), [14](#)
- kobo_form, [15](#)
- kobo_lang, [16](#)
- kobo_lang.character (kobo_lang), [16](#)
- kobo_lang.default (kobo_lang), [16](#)
- kobo_lang.kobo_asset (kobo_lang), [16](#)
- kobo_lang_get (kobo_lang), [16](#)
- kobo_lang_set (kobo_lang), [16](#)
- kobo_settings, [18](#)
- kobo_setup, [19](#)
- kobo_submissions (kobo_data), [12](#)
- kobo_token, [20](#)