

Package: ckanr (via r-universe)

August 30, 2026

Title Client for the Comprehensive Knowledge Archive Network ('CKAN') API

Description Client for 'CKAN' API (<<https://ckan.org/>>). Includes interface to 'CKAN' 'APIs' for search, list, show for packages, organizations, and resources. In addition, provides an interface to the 'datastore' API.

Version 0.8.1

License MIT + file LICENSE

LazyData true

URL <https://docs.ropensci.org/ckanr/> (website)
<https://github.com/ropensci/ckanr> (devel)

BugReports <https://github.com/ropensci/ckanr/issues>

Encoding UTF-8

Language en-US

Roxygen list(markdown = TRUE)

Depends DBI (>= 0.3.1)

Imports methods, stats, utils, crul, jsonlite (>= 0.9.17), dplyr (>= 0.7.0), dbplyr, magrittr

Suggests arrow, sf, readr, readxl, testthat, xml2, rlang

RoxygenNote 7.3.3

X-schema.org-keywords database, open-data, ckan, api, data, dataset

X-schema.org-applicationCategory Data Access

X-schema.org-isPartOf ``<https://ropensci.org>``

Config/pak/sysreqs libicu-dev libssl-dev

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

Date/Publication 2025-12-10 08:07:21 UTC

RemoteUrl <https://github.com/ropensci/ckanr>

RemoteRef main

RemoteSha 208b71c2a69876b94b3982918c4c675f27a37560

Contents

ckanr-package	4
activity_helpers	6
api_tokens	8
as.ckan_group	10
as.ckan_organization	11
as.ckan_package	11
as.ckan_related	12
as.ckan_resource	13
as.ckan_resource_view	14
as.ckan_tag	15
as.ckan_user	16
as.ckan_vocabulary	16
changes	17
ckan_action	18
ckan_classes	19
ckan_fetch	20
ckan_info	23
ckanr-deprecated	24
ckanr_settings	24
ckanr_setup	25
config_options	27
dashboard_activity_list	28
dashboard_count	29
dataset_purge	30
diagnostics	31
ds_create	32
ds_create_dataset	33
ds_search	35
ds_search_sql	37
follow_objects	38
follow_users	39
followee_counts	40
followee_lists	41
follower_counts	43
follower_lists	44
group_create	45
group_delete	47
group_list	48
group_list_authz	49
group_member_create	50
group_member_delete	51
group_patch	52
group_show	53
group_update	54
job_queue	55
license_list	56

member_create	57
member_delete	58
member_list	59
member_roles_list	60
membership	60
organization_create	61
organization_delete	62
organization_list	63
organization_list_for_user	64
organization_member_create	65
organization_member_delete	66
organization_purge	67
organization_show	68
package_activity_list	69
package_collaborator_create	70
package_collaborator_delete	71
package_collaborator_list	72
package_collaborator_list_for_user	73
package_collaborators	73
package_create	74
package_create_default_resource_views	76
package_dataset_extras	77
package_delete	77
package_list	78
package_list_current	79
package_owner_org_update	80
package_patch	80
package_relationship_create	82
package_relationship_delete	83
package_relationship_update	83
package_relationships	84
package_relationships_list	85
package_resource_reorder	86
package_revise	87
package_revision_list	88
package_search	89
package_show	91
package_update	92
ping	93
related_create	94
related_delete	96
related_list	97
related_show	98
resource_create	99
resource_create_default_resource_views	101
resource_delete	102
resource_patch	103
resource_search	104

resource_show	105
resource_update	107
resource_view_clear	110
resource_view_create	110
resource_view_delete	111
resource_view_list	112
resource_view_reorder	112
resource_view_show	113
resource_view_update	114
resource_views	115
revision_list	115
servers	116
src_ckan	116
tag_autocomplete	117
tag_create	118
tag_list	119
tag_search	120
tag_show	121
task_status	122
term_translation	124
user_activity_list	125
user_create	127
user_delete	128
user_invite	129
user_list	130
user_show	131
vocabulary	132
Index	135

 ckanr-package

R client for the CKAN API

Description

ckanr is a full client for the CKAN API, wrapping all APIs, including for reading and writing data. Please get in touch (<https://github.com/ropensci/ckanr/issues> or <https://discuss.ropensci.org/>) if you have problems, or have use cases that we don't cover yet.

CKAN API

Documentation for the CKAN API is at <https://docs.ckan.org/en/latest/api/index.html>. We'll always be following the latest version of the API.

ckanr package API

The functions can be grouped into those for setup, packages, resources, tags, organizations, groups, and users.

- Setup - The main one is `ckanr_setup()` - and many related functions, e.g., `get_default_key()`
- Packages - Create a package with `package_create()`, and see other functions starting with `package_*`
- Resources - Create a package with `resource_create()`, and see other functions starting with `resource_*`
- Tags - List tags with `tag_list()`, and see other functions starting with `tag_*`
- Organizations - List organizations with `organization_list()`, show a specific organization with `organization_show()`, and create with `organization_create()`
- Groups - List groups with `group_list()`, and see other functions starting with `group_*`
- Users - List users with `user_list()`, and see other functions starting with `user_*`
- Related items - See functions starting with `related_*`

Datastore

We are also working on supporting the Datastore extension (<https://docs.ckan.org/en/latest/maintaining/datastore.html>). We currently have these functions:

- `ds_create()`
- `ds_create_dataset()`
- `ds_search()`
- `ds_search_sql()`

Fetch

Data can come back in a huge variety of formats. We've attempted a function to help you fetch not just metadata but the actual data for a link to a file on a CKAN instance. Though if you know what you're doing, you can easily use whatever is your preferred tool for the job (e.g., maybe you like `read.csv()` for reading csv files).

CKAN Instances

We have a helper function (`servers()`) that spits out the current CKAN instances we know about, with URLs to their base URLs that should work using this package. That is, not necessarily landing pages of each instance, although, the URL may be the landing page and the base API URL.

Author(s)

Scott Chamberlain <myrmecocystus@gmail.com>

Florian Mayer <florian.wendelin.mayer@gmail.com>

Wush Wu

Immanuel Costigan <i.costigan@me.com>

Sharla Gelfand

Francisco Alves <fjunior.alves.oliveira@gmail.com>

See Also

Useful links:

- [https://docs.ropensci.org/ckanr/\(website\)https://github.com/ropensci/ckanr\(devel\)](https://docs.ropensci.org/ckanr/(website)https://github.com/ropensci/ckanr(devel))
- Report bugs at <https://github.com/ropensci/ckanr/issues>

activity_helpers

Activity stream helpers

Description

These helpers wrap CKAN's `ckanext.activity` endpoints. They require the activity core plugin to be enabled on the target CKAN instance.

Usage

```
group_activity_list(  
  id,  
  offset = 0,  
  limit = 31,  
  include_hidden_activity = FALSE,  
  url = get_default_url(),  
  key = get_default_key(),  
  as = "list",  
  ...  
)  
  
organization_activity_list(  
  id,  
  offset = 0,  
  limit = 31,  
  include_hidden_activity = FALSE,  
  url = get_default_url(),  
  key = get_default_key(),  
  as = "list",  
  ...  
)  
  
recently_changed_packages_activity_list(  
  offset = 0,  
  limit = 31,  
  url = get_default_url(),  
  key = get_default_key(),  
  as = "list",  
  ...  
)
```

```
dashboard_new_activities_count(  
    url = get_default_url(),  
    key = get_default_key(),  
    as = "list",  
    ...  
)  
  
dashboard_mark_activities_old(  
    url = get_default_url(),  
    key = get_default_key(),  
    as = "list",  
    ...  
)  
  
activity_show(  
    id,  
    url = get_default_url(),  
    key = get_default_key(),  
    as = "list",  
    ...  
)  
  
activity_data_show(  
    id,  
    object_type,  
    url = get_default_url(),  
    key = get_default_key(),  
    as = "list",  
    ...  
)  
  
activity_diff(  
    id,  
    object_type,  
    diff_type = "unified",  
    url = get_default_url(),  
    key = get_default_key(),  
    as = "list",  
    ...  
)  
  
activity_create(  
    user_id,  
    object_id,  
    activity_type,  
    data = NULL,  
    url = get_default_url(),
```

```

    key = get_default_key(),
    as = "list",
    ...
)

send_email_notifications(
    url = get_default_url(),
    key = get_default_key(),
    as = "list",
    ...
)

```

Arguments

id	(character) Identifier of the target object (group, organization, activity, etc.).
offset	(numeric) Where to start getting activity items from (optional, default: 0)
limit	(numeric) The maximum number of activities to return (optional, default: 31)
include_hidden_activity	(logical) If TRUE, include private activity entries (requires sysadmin).
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)
object_type	(character) Domain object affected by the activity, e.g., "package", "resource", or a plugin-defined type.
diff_type	(character) Diff format returned by <code>activity_diff()</code> , typically "unified".
user_id	(character) User identifier associated with a custom <code>activity_create()</code> entry.
object_id	(character) Target object identifier for <code>activity_create()</code> .
activity_type	(character) Activity type string to emit via <code>activity_create()</code> .
data	(list character) Optional structured payload describing the activity body. Lists are JSON-encoded automatically.

api_tokens

API token helpers

Description

Inspect, create, and revoke API tokens for a user.

Usage

```

api_token_list(
  user_id = NULL,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)

api_token_create(
  user,
  name,
  extra_fields = NULL,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)

api_token_revoke(
  token = NULL,
  jti = NULL,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)

```

Arguments

<code>user_id</code>	(character) User identifier to filter tokens.
<code>url</code>	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
<code>key</code>	A privileged CKAN API key, Default: your key set with ckanr_setup
<code>as</code>	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to <code>data.frame</code> 's when possible, so the result can vary from a vector, list or <code>data.frame</code> . (required)
<code>...</code>	Curl args passed on to verb-POST (optional)
<code>user</code>	(character) Name or id of the token owner.
<code>name</code>	(character) Friendly token name.
<code>extra_fields</code>	(list) Optional named list of additional fields accepted by CKAN or extensions.
<code>token</code>	(character) Encoded API token value.
<code>jti</code>	(character) Token identifier (overrides token when provided).

Examples

```
## Not run:
ckanr_setup(url = "https://demo.ckan.org/", key = "my-ckan-key")
api_token_list(user_id = "my-user-id")

## End(Not run)
```

as.ckan_group

ckan_group class helpers

Description

ckan_group class helpers

Usage

```
as.ckan_group(x, ...)
```

```
is.ckan_group(x)
```

Arguments

x	Variety of things, character, list, or ckan_group class object
...	Further args passed on to group_show() if character given

Examples

```
## Not run:
ckanr_setup(url = "https://demo.ckan.org/", key = getOption("ckan_demo_key"))

(grps <- group_list())
grps[[3]]

# create item class from only an item ID
as.ckan_group(grps[[3]]$id)

# gives back itself
(x <- as.ckan_group(grps[[3]]$id))
as.ckan_group(x)

## End(Not run)
```

as.ckan_organization *ckan_organization class helpers*

Description

ckan_organization class helpers

Usage

```
as.ckan_organization(x, ...)
```

```
is.ckan_organization(x)
```

Arguments

x	Variety of things, character, list, or ckan_organization class object
...	Further args passed on to organization_show() if character given

Examples

```
## Not run:
ckanr_setup(
  url = "https://demo.ckan.org/",
  key = getOption("ckan_demo_key")
)

(orgs <- organization_list(limit = 3))
orgs[[3]]

# create item class from only an item ID
as.ckan_organization(orgs[[3]]$id)

# gives back itself
(x <- as.ckan_organization(orgs[[3]]$id))
as.ckan_organization(x)

## End(Not run)
```

as.ckan_package *ckan_package class helpers*

Description

ckan_package class helpers

Usage

```
as.ckan_package(x, ...)
```

```
is.ckan_package(x)
```

Arguments

x	Variety of things, character, list, or ckan_package class object
...	Further args passed on to package_show() if character given. In particular, if GET is not supported you can try the http_method parameter to set a different HTTP verb

Examples

```
## Not run:
ckanr_setup(
  url = "https://demo.ckan.org/",
  key = getOption("ckan_demo_key")
)

(pkgs <- package_search())
pkgs$results
pkgs$results[[3]]

# create item class from only an item ID
as.ckan_package(pkgs$results[[3]]$id)

# gives back itself
(x <- as.ckan_package(pkgs$results[[3]]$id))
as.ckan_package(x)

## End(Not run)
```

as.ckan_related	<i>ckan_related class helpers</i>
-----------------	-----------------------------------

Description

ckan_related class helpers

Usage

```
as.ckan_related(x, ...)
```

```
is.ckan_related(x)
```

Arguments

`x` Variety of things, character, list, or ckan_related class object
`...` Further args passed on to [related_show\(\)](#) if character given

Examples

```
## Not run:
ckanr_setup(
  url = "https://demo.ckan.org/",
  key = getOption("ckan_demo_key")
)

(x <- package_create("foobbbbbarrrrr") %>%
  related_create(
    title = "my resource",
    type = "visualization"
  ))

# create item class from only an item ID
as.ckan_related(x$id)

# gives back itself
(x <- as.ckan_related(x$id))
as.ckan_related(x)

## End(Not run)
```

as.ckan_resource	<i>ckan_resource class helpers</i>
------------------	------------------------------------

Description

ckan_resource class helpers

Usage

```
as.ckan_resource(x, ...)
```

```
is.ckan_resource(x)
```

Arguments

`x` Variety of things, character, list, or ckan_package class object
`...` Further args passed on to [resource_show\(\)](#) if character given

Examples

```
## Not run:
ckanr_setup(
  url = "https://demo.ckan.org/",
  key = getOption("ckan_demo_key")
)

(resrcs <- resource_search(q = "name:data"))
resrcs$results
resrcs$results[[3]]

# create item class from only an item ID
as.ckan_resource(resrcs$results[[3]]$id)

# gives back itself
(x <- as.ckan_resource(resrcs$results[[3]]$id))
as.ckan_resource(x)

## End(Not run)
```

as.ckan_resource_view *ckan_resource_view class helpers*

Description

ckan_resource_view class helpers

Usage

```
as.ckan_resource_view(x, ...)

is.ckan_resource_view(x)
```

Arguments

x	Variety of things: character, list, or ckan_resource_view object
...	Further arguments passed to resource_view_show() when x is an identifier.

Examples

```
## Not run:
ckanr_setup(
  url = "https://demo.ckan.org/",
  key = Sys.getenv("CKAN_DEMO_KEY")
)

res <- package_show("sample-dataset")
views <- resource_view_list(res$resources[[1]]$id)
```

```
# Coerce from ID
as.ckan_resource_view(views[[1]]$id)

# Pass through existing objects
as.ckan_resource_view(views[[1]])

## End(Not run)
```

as.ckan_tag

ckan_tag class helpers

Description

ckan_tag class helpers

Usage

```
as.ckan_tag(x, ...)

is.ckan_tag(x)
```

Arguments

x	Variety of things, character, list, or ckan_tag class object
...	Further args passed on to tag_show() if character given

Examples

```
## Not run:
ckanr_setup(
  url = "https://demo.ckan.org/",
  key = getOption("ckan_demo_key")
)

(tags <- tag_search(query = "ta"))
tags[[3]]

# create item class from only an item ID
as.ckan_tag(tags[[3]]$id)

# gives back itself
(x <- as.ckan_tag(tags[[3]]$id))
as.ckan_tag(x)

## End(Not run)
```

as.ckan_user *ckan_user class helpers*

Description

ckan_user class helpers

Usage

```
as.ckan_user(x, ...)
```

```
is.ckan_user(x)
```

Arguments

x	Variety of things, character, list, or ckan_user class object
...	Further args passed on to user_show() if character given

Examples

```
## Not run:
ckanr_setup(
  url = "https://demo.ckan.org/",
  key = getOption("ckan_demo_key")
)

(usr <- user_list())
usr[1:3]
usr[[3]]

# create item class from only an item ID
as.ckan_user(usr[[3]]$id)

# gives back itself
(x <- as.ckan_user(usr[[3]]$id))
as.ckan_user(x)

## End(Not run)
```

as.ckan_vocabulary *ckan_vocabulary class helpers*

Description

ckan_vocabulary class helpers

Usage

```
as.ckan_vocabulary(x, ...)
```

```
is.ckan_vocabulary(x)
```

Arguments

x	Variety of inputs: character, list, or ckan_vocabulary object
...	Additional arguments passed to vocabulary_show() when retrieving by identifier.

Examples

```
## Not run:
ckanr_setup(url = "https://demo.ckan.org/", key = getOption("ckan_demo_key"))

vocab <- vocabulary_create(name = sprintf("demo_vocab_%s", sample.int(1e4, 1)))

# create class from an id
as.ckan_vocabulary(vocab$id)

# passing through existing objects is a no-op
as.ckan_vocabulary(vocab)

## End(Not run)
```

changes

Get an activity stream of recently changed datasets on a site.

Description

Get an activity stream of recently changed datasets on a site.

Usage

```
changes(
  offset = 0,
  limit = 31,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

offset	(numeric) Where to start getting activity items from (optional, default: 0)
limit	(numeric) The maximum number of activities to return (optional, default: 31)
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to <code>data.frame</code> 's when possible, so the result can vary from a vector, list or <code>data.frame</code> . (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
changes()
changes(as = "json")
changes(as = "table")

## End(Not run)
```

ckan_action

Generic action function

Description

You have to set the CKAN action to use, as well as the HTTP verb, in addition to any query parameters, request body data, and request headers.

Usage

```
ckan_action(
  action,
  query = NULL,
  body = NULL,
  headers = list(),
  verb = "POST",
  url = get_default_url(),
  key = get_default_key(),
  ...
)
```

Arguments

action	A valid CKAN API action name (e.g., "package_list", "package_show"). See the CKAN API documentation for a full list of actions: https://docs.ckan.org/en/latest/api/index.html
query	a named list of URL query parameters
body	data to be used in the body of a request, see https://docs.ropensci.org/crul/reference/verb-POST.html for options
headers	a named list of request headers
verb	HTTP request verb, e.g., GET, POST
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup() and get_default_url()
key	A privileged CKAN API key, Default: your key set with ckanr_setup
...	Curl args passed on to the relevant crul::HttpClient method depending on the verb parameter (optional)

Value

a text string, because we can't know ahead of time for sure what kind of data will be returned, we return text and the user can parse as needed.

Examples

```
## Not run:
ckanr_setup(
  url = "https://demo.ckan.org/",
  key = getOption("ckan_demo_key")
)

ckan_action("package_list")
ckan_action("package_list", verb = "GET")
ckan_action("package_list", url = "https://data.nhm.ac.uk")

## End(Not run)
```

ckan_classes

ckanr S3 classes

Description

ckanr S3 classes

The classes

- `ckan_package` - CKAN package
- `ckan_resource` - CKAN resource
- `ckan_resource_view` - CKAN resource view
- `ckan_related` - CKAN related item

Coercion

The functions `as.ckan_*`() for each CKAN object type coerce something to a S3 class of that type. For example, you can coerce a package ID as a character string into an `ckan_package` object by calling `as.ckan_package(<id>)`.

Testing for classes

To test whether an object is of a particular `ckan_*` class, there is a `is._ckan_*`() function for all of the classes listed above. You can use one of those functions to get a logical back, TRUE or FALSE.

Manipulation

These are simple S3 classes, basically an R list with an attached class so we can know what to do with the object and have flexible inputs and outputs from functions. You can edit one of these classes yourself by simply changing values in the list.

ckan_fetch

Download a file

Description

Download a file

Usage

```
ckan_fetch(
  x,
  store = "session",
  path = "file",
  format = NULL,
  key = get_default_key(),
  ...
)
```

Arguments

<code>x</code>	URL for the file
<code>store</code>	One of session (default) or disk. session stores in R session, and disk saves the file to disk.
<code>path</code>	if store="disk", you must give a path to store file to
<code>format</code>	Format of the file. Required if format is not detectable through file URL.
<code>key</code>	A CKAN API key (optional, character)
<code>...</code>	Arguments passed on to the function used to read the file, if store="session". See details.

Details

The ... argument can be used to pass additional arguments to the function that actually reads the file. It is only used when store="session" and if the file is *not* a ZIP file.

This list shows what function is used to read what file format, so that you can see what additional arguments are available:

- csv: [read.csv](#)
- xls, xlsx: [read_excel](#)
- xml: [read_xml](#)
- html: [read_html](#)
- json: [fromJSON](#)
- shp, geojson: [st_read](#)
- txt: [read.table](#)
- parquet: [read_parquet](#)

Examples

```
## Not run:
# CSV file
ckanr_setup("http://datamx.io")
res <- resource_show(
  id = "6145a539-cbde-4b0d-a3d3-d1a5eb013f5c",
  as = "table"
)
head(ckan_fetch(res$url))
ckan_fetch(res$url, "disk", "myfile.csv")

# CSV file, format not available
ckanr_setup("https://ckan0.cf.opendata.inter.prod-toronto.ca")
res <- resource_show(
  id = "c57c3e1c-20e2-470f-bc82-e39a0264be31",
  as = "table"
)
res$url
res$format
head(ckan_fetch(res$url, format = res$format))

# Excel file - requires readxl package
ckanr_setup("http://datamx.io")
res <- resource_show(
  id = "e883510e-a082-435c-872a-c5b915857ae1",
  as = "table"
)
head(ckan_fetch(res$url))

# Excel file, multiple sheets - requires readxl package
ckanr_setup()
res <- resource_show(
  id = "ce02a1cf-35f1-41df-91d9-11ed1fdd4186",
```

```

    as = "table"
  )
  x <- ckan_fetch(res$url)
  names(x)
  head(x[["Mayor - Maire"]])

# XML file - requires xml2 package
# ckanr_setup("http://data.ottawa.ca")
# res <- resource_show(id = "380061c1-6c46-4da6-a01b-7ab0f49a881e",
# as = "table")
# ckan_fetch(res$url)

# HTML file - requires xml2 package
ckanr_setup("http://open.canada.ca/data/en")
res <- resource_show(
  id = "80321bac-4283-487c-93bd-c65acaa660f5",
  as = "table"
)
ckan_fetch(res$url)
library("xml2")
xml_text(xml_find_first(xml_children(ckan_fetch(res$url))[[1]], "title"))

# JSON file, by default reads in to a data.frame for ease of use
ckanr_setup("http://data.surrey.ca")
res <- resource_show(
  id = "8d07c662-800d-4977-9e3e-5a3d2d1e99ab",
  as = "table"
)
head(ckan_fetch(res$url))

# SHP file (spatial data, ESRI format) - requires sf package
ckanr_setup("https://ckan0.cf.opendata.inter.prod-toronto.ca")
res <- resource_show(
  id = "27362290-8bbf-434b-a9de-325a6c2ef923",
  as = "table"
)
x <- ckan_fetch(res$url)
class(x)
plot(x[, c("AREA_NAME", "geometry")])

# GeoJSON file - requires sf package
ckanr_setup("http://datamx.io")
res <- resource_show(
  id = "b1cd35b7-479e-4fa0-86e9-e897d3c617e6",
  as = "table"
)
x <- ckan_fetch(res$url)
class(x)
plot(x[, c("mun_name", "geometry")])

# ZIP file - packages required depends on contents
ckanr_setup("https://ckan0.cf.opendata.inter.prod-toronto.ca")
res <- resource_show(

```

```

    id = "bb21e1b8-a466-41c6-8bc3-3c362cb1ed55",
    as = "table"
  )
  x <- ckan_fetch(res$url)
  names(x)
  head(x[["ChickenpoxAgegroups2017.csv"]])

# TXT file
ckanr_setup("https://ckan0.cf.opendata.inter.prod-toronto.ca")
res <- resource_show(
  id = "e4211f49-611f-438c-a444-aaa7f3f84117",
  as = "table"
)
x <- ckan_fetch(res$url)
head(x)

# TXT file, semicolon used as separator
ckanr_setup("https://data.coat.no")
res <- resource_show(
  id = "384fe537-e0bd-4e57-8a0d-420b7a745196",
  as = "table"
)
x <- ckan_fetch(res$url, sep = ";")
head(x)

# Parquet file - requires arrow package
ckanr_setup("https://data.coat.no")
res <- resource_show(
  id = "b7d1c192-0f44-4bab-bd51-eea0d408a09f",
  as = "table"
)
x <- ckan_fetch(res$url)
head(x)

## End(Not run)

```

ckan_info

Get information on a CKAN server

Description

Get information on a CKAN server

Usage

```
ckan_info(url = get_default_url(), ...)
```

```
ckan_version(url = get_default_url(), ...)
```

Arguments

url	Base url to use. Default: https://data.ontario.ca . See also ckanr_setup() and get_default_url() . (required)
...	Curl args passed on to crul::verb-GET (optional)

Value

for `ckan_info` a list with many slots with various info. for `ckan_version`, list of length two, with actual version as character, and another with version converted to numeric (any dots or letters removed)

Examples

```
## Not run:
ckan_info()
ckan_info(servers()[5])

ckan_version(servers()[5])

## End(Not run)
```

ckanr-deprecated	<i>Deprecated functions in ckanr</i>
------------------	---

Description

These functions still work but will be removed (defunct) in the next version.

Details

- [ds_create_dataset\(\)](#): The functionality of this function is already in another function in this package. See function [resource_create\(\)](#)

ckanr_settings	<i>Get or set ckanr CKAN settings</i>
----------------	---------------------------------------

Description

Get or set ckanr CKAN settings

Usage

```
ckanr_settings()

get_default_url()

get_default_key()

get_test_url()

get_test_key()

get_test_did()

get_test_rid()

get_test_gid()

get_test_oid()

get_test_behaviour()
```

Value

ckanr_settings prints your base url, API key (if used), and optional test server settings (URL, API key, a dataset ID and a resource ID). ckanr_setup sets your production and test settings, while get_test_* get each of those respective settings. test_behaviour indicates whether the CKANR test suite will skip ("SKIP") or fail ("FAIL") writing tests in case the configured test CKAN settings don't work.

See Also

[ckanr_setup\(\)](#), [get_default_url\(\)](#), [get_default_key\(\)](#), [get_test_url\(\)](#), [get_test_key\(\)](#), [get_test_did\(\)](#), [get_test_rid\(\)](#), [get_test_gid\(\)](#), [get_test_oid](#), [get_test_behaviour](#)

Examples

```
ckanr_settings()
```

ckanr_setup	<i>Configure default CKAN settings</i>
-------------	--

Description

Configure default CKAN settings

Usage

```
ckanr_setup(
  url = "https://demo.ckan.org/",
  key = NULL,
  test_url = NULL,
  test_key = NULL,
  test_did = NULL,
  test_rid = NULL,
  test_gid = NULL,
  test_oid = NULL,
  test_behaviour = NULL,
  proxy = NULL
)
```

Arguments

url	A CKAN URL (optional), default: https://data.ontario.ca
key	A CKAN API key (optional, character)
test_url	(optional, character) A valid CKAN URL for testing purposes
test_key	(optional, character) A valid CKAN API key privileged to create datasets at test_url
test_did	(optional, character) A valid CKAN dataset ID, existing at test_url
test_rid	(optional, character) A valid CKAN resource ID, attached to did
test_gid	(optional, character) A valid CKAN group name at test_url
test_oid	(optional, character) A valid CKAN organization name at test_url
test_behaviour	(optional, character) Whether to fail ("FAIL") or skip ("SKIP") writing tests in case of problems with the configured test CKAN.
proxy	an object of class request from a call to crul::proxy()

Details

[ckanr_setup\(\)](#) sets CKAN connection details. ckanr's functions default to use the default URL and API key unless specified explicitly.

ckanr's automated tests require a valid CKAN URL, a privileged API key for that URL, plus the IDs of an existing dataset and an existing resource, respectively.

The writing tests (create, update, delete) can fail for two reasons: failures in ckanr's code which the tests aim to detect, or failures in the configured CKAN, which are not necessarily a problem with ckanr's code but prevent the tests to prove otherwise.

Setting test_behaviour to "SKIP" will allow writing tests to skip if the configured test CKAN fails. This is desirable to e.g. test the other functions even if the tester has no write access to a CKAN instance.

Setting test_behaviour to "FAIL" will let the tester find any problems with both the configured test CKAN and the writing functions.

Examples

```
# CKAN users without admin/editor privileges could run:
ckanr_setup(url = "https://demo.ckan.org/")

# Privileged CKAN editor/admin users can run:
ckanr_setup(url = "https://demo.ckan.org/", key = "some-CKAN-API-key")

# ckanR developers/testers can either use the Codespaces/devcontainer test suite
# which automatically sets up all test credentials, or run:
ckanr_setup(
  url = "https://demo.ckan.org/", key = "some-CKAN-API-key",
  test_url = "http://test-ckan.gov/", test_key = "test-ckan-API-key",
  test_did = "test-ckan-dataset-id", test_rid = "test-ckan-resource-id",
  test_gid = "test-group-name", test_oid = "test-organization-name",
  test_behaviour = "FAIL"
)

# Not specifying the default CKAN URL will reset the CKAN URL to its default
# "https://demo.ckan.org/":
ckanr_setup()

# set a proxy
ckanr_setup(proxy = crul::proxy("64.251.21.73:8080"))
ckanr_settings()
## run without setting proxy to reset to no proxy
ckanr_setup()
ckanr_settings()
```

config_options

Runtime configuration helpers

Description

Inspect and update CKAN's runtime-editable configuration keys.

Usage

```
config_option_list(
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)

config_option_show(
  option_key,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
```

```

    ...
  )

  config_option_update(
    options,
    url = get_default_url(),
    key = get_default_key(),
    as = "list",
    ...
  )

```

Arguments

url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to <code>data.frame</code> 's when possible, so the result can vary from a vector, list or <code>data.frame</code> . (required)
...	Curl args passed on to verb-POST (optional)
option_key	(character) Configuration option identifier.
options	(list) Named list of configuration options to update.

Examples

```

## Not run:
ckanr_setup(url = "https://demo.ckan.org/", key = "my-ckan-key")
config_option_list()
config_option_update(list("ckan.site_title" = "My Portal"))

## End(Not run)

```

dashboard_activity_list

Authorized user's dashboard activity stream

Description

Authorized user's dashboard activity stream

Usage

```
dashboard_activity_list(
  limit = 31,
  offset = 0,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

limit	(integer) The maximum number of activities to return (optional). Default: 31
offset	(integer) Where to start getting activity items from (optional). Default: 0
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to <code>data.frame</code> 's when possible, so the result can vary from a vector, list or <code>data.frame</code> . (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
# Setup
ckanr_setup(url = "https://demo.ckan.org/", key = getOption("ckan_demo_key"))

# get activity
(res <- dashboard_activity_list())

## End(Not run)
```

dashboard_count	<i>Number of new activities of an authorized user</i>
-----------------	---

Description

Number of new activities of an authorized user

Usage

```
dashboard_count(
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

Details

Activities from the user herself are not counted by CKAN even though they appear in the dashboard (users don't want to be notified about things they did themselves).

Examples

```
## Not run:
# Setup
ckanr_setup(url = "https://demo.ckan.org/", key = getOption("ckan_demo_key"))

# count
dashboard_new_activities_count()

## End(Not run)
```

dataset_purge	<i>Permanently purge a dataset</i>
---------------	------------------------------------

Description

Permanently purge a dataset

Usage

```
dataset_purge(id, url = get_default_url(), key = get_default_key(), ...)
```

Arguments

<code>id</code>	(character or <code>ckan_package</code>) Dataset identifier.
<code>url</code>	Base url to use. Default: <code>https://demo.ckan.org/</code> See also ckanr_setup and get_default_url .
<code>key</code>	A privileged CKAN API key, Default: your key set with ckanr_setup
<code>...</code>	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
dataset_purge("dataset-id")

## End(Not run)
```

diagnostics

Diagnostics helpers

Description

Lightweight wrappers for CKAN's `status_show` and `help_show` actions.

Usage

```
status_show(url = get_default_url(), key = get_default_key(), as = "list", ...)

help_show(
  name,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

<code>url</code>	Base url to use. Default: <code>https://demo.ckan.org/</code> See also ckanr_setup and get_default_url .
<code>key</code>	A privileged CKAN API key, Default: your key set with ckanr_setup
<code>as</code>	(character) One of <code>list</code> (default), <code>table</code> , or <code>json</code> . Parsing with <code>table</code> option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to <code>data.frame</code> 's when possible, so the result can vary from a vector, list or <code>data.frame</code> . (required)
<code>...</code>	Curl args passed on to verb-POST (optional)
<code>name</code>	(character) CKAN action name to describe.

Examples

```
## Not run:
ckanr_setup(url = "https://demo.ckan.org/")
status_show()
help_show("package_search")

## End(Not run)
```

ds_create

Add a new table to a datastore

Description

BEWARE: This function still doesn't quite work yet.

Usage

```
ds_create(
  resource_id = NULL,
  resource = NULL,
  force = FALSE,
  aliases = NULL,
  fields = NULL,
  records = NULL,
  primary_key = NULL,
  indexes = NULL,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

resource_id	(string) Resource id that the data is going to be stored against.
resource	(dictionary) Resource dictionary that is passed to resource_create() . Use instead of resource_id (optional)
force	(logical) Set to TRUE to edit a read-only resource. Default: FALSE
aliases	(character) Names for read only aliases of the resource. (optional)
fields	(list) Fields/columns and their extra metadata. (optional)
records	(list) The data, eg: [{"dob": "2005", "some_stuff": ["a", "b"]}]] (optional)
primary_key	(character) Fields that represent a unique key (optional)
indexes	(character) Indexes on table (optional)

url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to <code>data.frame</code> 's when possible, so the result can vary from a vector, list or <code>data.frame</code> . (required)
...	Curl args passed on to verb-POST (optional)

References

http://bit.ly/ds_create

Examples

```
## Not run:
ckanr_setup(
  url = "https://demo.ckan.org/",
  key = getOption("ckan_demo_key")
)

# create a package
(res <- package_create("foobarrrrr", author = "Jane Doe"))

# then create a resource
file <- system.file("examples", "actinidiaceae.csv", package = "ckanr")
(xx <- resource_create(
  package_id = res$id,
  description = "my resource",
  name = "bears",
  upload = file,
  rcurl = "http://google.com"
))
ds_create(resource_id = xx$id, records = iris, force = TRUE)
resource_show(xx$id)

## End(Not run)
```

ds_create_dataset

Datastore - create a new resource on an existing dataset

Description

Datastore - create a new resource on an existing dataset

Usage

```
ds_create_dataset(
  package_id,
  name,
  path,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

package_id	(character) Existing package ID (required)
name	(character) Name of the new resource (required)
path	(character) Path of the file to add (required)
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

Details

This function is deprecated - will be defunct in the next version of this package

References

http://docs.ckan.org/en/latest/api/index.html#ckan.logic.action.create.resource_create

Examples

```
## Not run:
path <- system.file("examples", "actinidiaceae.csv", package = "ckanr")
ckanr_setup(url = "https://demo.ckan.org/", key = "my-demo-ckan-org-api-key")
ds_create_dataset(package_id = "testingagain", name = "mydata", path = path)

# Testing: see ?ckanr_setup to set test settings
ckanr_setup(
  test_url = "http://my-ckan.org/",
  test_key = "my-ckan-api-key",
  test_did = "an-existing-package-id",
  test_rid = "an-existing-resource-id"
)
ds_create_dataset(
```

```

package_id = get_test_pid(), name = "mydata",
path = system.file("examples",
  "actinidiaceae.csv",
  package = "ckanr"
),
key = get_test_key(),
url = get_test_url()
)

## End(Not run)

```

ds_search

*Datastore - search or get a dataset from CKAN datastore***Description**

Datastore - search or get a dataset from CKAN datastore

Usage

```

ds_search(
  resource_id = NULL,
  filters = NULL,
  q = NULL,
  plain = NULL,
  language = NULL,
  fields = NULL,
  offset = NULL,
  limit = NULL,
  sort = NULL,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)

```

Arguments

resource_id	(character) id or alias of the resource to be searched against
filters	(character) matching conditions to select, e.g {"key1": "a", "key2": "b"} (optional)
q	(character) full text query (optional)
plain	(character) treat as plain text query (optional, default: TRUE)
language	(character) language of the full text query (optional, default: english)
fields	(character) fields to return (optional, default: all fields in original order)
offset	(numeric) Where to start getting activity items from (optional, default: 0)

limit	(numeric) The maximum number of activities to return (optional, default: 100)
sort	Field to sort on. You can specify ascending (e.g., score desc) or descending (e.g., score asc), sort by two fields (e.g., score desc, price asc), or sort by a function (e.g., sum(x_f, y_f) desc, which sorts by the sum of x_f and y_f in a descending order). (optional)
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to <code>data.frame</code> 's when possible, so the result can vary from a vector, list or <code>data.frame</code> . (required)
...	Curl args passed on to verb-POST (optional)

Details

From the help for this method "The `datastore_search` action allows you to search data in a resource. DataStore resources that belong to private CKAN resource can only be read by you if you have access to the CKAN resource and send the appropriate authorization."

Setting `plain=FALSE` enables the entire PostgreSQL *full text search query language*. A listing of all available resources can be found at the alias *table_metadata* full text search query language: <http://www.postgresql.org/docs/9.1/static/datatype-textsearch.html#DATATYPE-TSQUERY>

Examples

```
## Not run:
ckanr_setup(url = "https://data.nhm.ac.uk/")

ds_search(resource_id = "8f0784a6-82dd-44e7-b105-6194e046eb8d")
ds_search(
  resource_id = "8f0784a6-82dd-44e7-b105-6194e046eb8d",
  as = "table"
)
ds_search(
  resource_id = "8f0784a6-82dd-44e7-b105-6194e046eb8d",
  as = "json"
)

ds_search(
  resource_id = "8f0784a6-82dd-44e7-b105-6194e046eb8d", limit = 1,
  as = "table"
)
ds_search(resource_id = "8f0784a6-82dd-44e7-b105-6194e046eb8d", q = "a*")

## End(Not run)
```

ds_search_sql	<i>Datastore - search or get a dataset from CKAN datastore</i>
---------------	--

Description

Datastore - search or get a dataset from CKAN datastore

Usage

```
ds_search_sql(
  sql,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

sql	(character) A single SQL select statement. (required)
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
url <- "https://demo.ckan.org/"
sql <- 'SELECT * from "f4129802-22aa-4437-b9f9-8a8f3b7b2a53" LIMIT 2'
ds_search_sql(sql, url = url, as = "table")
sql2 <- 'SELECT "Species","Genus","Family" from "f4129802-22aa-4437-b9f9-8a8f3b7b2a53" LIMIT 2'
ds_search_sql(sql2, url = url, as = "table")

## End(Not run)
```

follow_objects*Follow and unfollow CKAN datasets or groups*

Description

Create or remove follower relationships for a CKAN object. These helpers require an authenticated user (API key) representing the follower.

Usage

```
dataset_follow(  
    id,  
    url = get_default_url(),  
    key = get_default_key(),  
    as = "list",  
    ...  
)  
  
dataset_unfollow(  
    id,  
    url = get_default_url(),  
    key = get_default_key(),  
    as = "list",  
    ...  
)  
  
dataset_am_following(id, url = get_default_url(), key = get_default_key(), ...)  
  
group_follow(  
    id,  
    url = get_default_url(),  
    key = get_default_key(),  
    as = "list",  
    ...  
)  
  
group_unfollow(  
    id,  
    url = get_default_url(),  
    key = get_default_key(),  
    as = "list",  
    ...  
)  
  
group_am_following(id, url = get_default_url(), key = get_default_key(), ...)
```

Arguments

id	(character or ckan_*) Identifier for the object to inspect. For datasets pass an id/slug or ckan_package, for groups pass the corresponding identifier or ckan_group, and for users pass a username or ckan_user.
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
ckanr_setup(url = "https://demo.ckan.org/", key = "my-key")
dataset_follow("my-dataset")
dataset_am_following("my-dataset")
dataset_unfollow("my-dataset")

## End(Not run)
```

follow_users	<i>Follow or unfollow users</i>
--------------	---------------------------------

Description

Manage user-to-user follower relationships.

Usage

```
follow_user(
  id,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)

unfollow_user(
  id,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

```
)

am_following_user(id, url = get_default_url(), key = get_default_key(), ...)
```

Arguments

id	(character or ckan_user) The user to follow or inspect.
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
ckanr_setup(url = "https://demo.ckan.org/", key = "my-key")
follow_user("demo-user")
am_following_user("demo-user")
unfollow_user("demo-user")

## End(Not run)
```

followee_counts	<i>Followee counts for CKAN users</i>
-----------------	---------------------------------------

Description

Count how many objects of each type (or overall) a user follows.

Usage

```
followee_count(
  id,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)

user_followee_count(
  id,
  url = get_default_url(),
  key = get_default_key(),
```

```

    as = "list",
    ...
)

dataset_followee_count(
  id,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)

group_followee_count(
  id,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)

```

Arguments

<code>id</code>	(character or <code>ckan_user</code>) User identifier.
<code>url</code>	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
<code>key</code>	A privileged CKAN API key, Default: your key set with ckanr_setup
<code>as</code>	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to <code>data.frame</code> 's when possible, so the result can vary from a vector, list or <code>data.frame</code> . (required)
<code>...</code>	Curl args passed on to verb-POST (optional)

Examples

```

## Not run:
ckanr_setup(url = "https://demo.ckan.org/", key = "my-key")
followee_count("demo-user")
user_followee_count("demo-user")
dataset_followee_count("demo-user")

## End(Not run)

```

Description

Return the objects being followed by a user.

Usage

```
followee_list(
  id,
  q = NULL,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)

user_followee_list(
  id,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)

dataset_followee_list(
  id,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)

group_followee_list(
  id,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

<code>id</code>	(character or ckan_user) User identifier.
<code>q</code>	(character) Optional prefix filter for <code>followee_list()</code> .
<code>url</code>	Base url to use. Default: <code>https://demo.ckan.org/</code> See also ckanr_setup and get_default_url .
<code>key</code>	A privileged CKAN API key, Default: your key set with ckanr_setup
<code>as</code>	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to

... parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)

... Curl args passed on to [verb-POST](#) (optional)

Examples

```
## Not run:
ckanr_setup(url = "https://demo.ckan.org/", key = "my-key")
followee_list("demo-user", q = "ckan")
dataset_followee_list("demo-user")

## End(Not run)
```

follower_counts	<i>Follower counts for CKAN objects</i>
-----------------	---

Description

Report follower totals for datasets, groups, or users.

Usage

```
dataset_follower_count(
  id,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)

group_follower_count(
  id,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)

user_follower_count(
  id,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

id	(character or ckan_*) Identifier for the object to inspect. For datasets pass an id/slug or ckan_package, for groups pass the corresponding identifier or ckan_group, and for users pass a username or ckan_user.
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
ckanr_setup(url = "https://demo.ckan.org/", key = "my-key")
dataset_follower_count("my-dataset")
group_follower_count("my-group")
user_follower_count("demo-user")

## End(Not run)
```

follower_lists

Follower lists for CKAN objects

Description

Return the individual follower records for CKAN datasets, groups, or users.

Usage

```
dataset_follower_list(
  id,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)

group_follower_list(
  id,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

```

)

user_follower_list(
  id,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)

```

Arguments

id	(character or ckan_*) Identifier for the object to inspect. For datasets pass an id/slug or ckan_package, for groups pass the corresponding identifier or ckan_group, and for users pass a username or ckan_user.
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

Examples

```

## Not run:
ckanr_setup(url = "https://demo.ckan.org/", key = "my-key")
dataset_follower_list("my-dataset")
group_follower_list("my-group")
user_follower_list("demo-user")

## End(Not run)

```

group_create

Create a group

Description

Create a group

Usage

```

group_create(
  name = NULL,
  id = NULL,
  title = NULL,

```

```

description = NULL,
image_url = NULL,
type = NULL,
state = "active",
approval_status = NULL,
extras = NULL,
packages = NULL,
groups = NULL,
users = NULL,
url = get_default_url(),
key = get_default_key(),
as = "list",
...
)

```

Arguments

name	(character) the name of the new dataset, must be between 2 and 100 characters long and contain only lowercase alphanumeric characters, • and _, e.g. 'warandpeace'
id	(character) The id of the group (optional)
title	(character) The title of the dataset (optional, default: same as name)
description	(character) The description of the group (optional)
image_url	(character) The URL to an image to be displayed on the group's page (optional)
type	(character) The type of the dataset (optional), IDatasetForm plugins associate themselves with different dataset types and provide custom dataset handling behaviour for these types
state	(character) The current state of the dataset, e.g. 'active' or 'deleted', only active datasets show up in search results and other lists of datasets, this parameter will be ignored if you are not authorized to change the state of the dataset (optional, default: 'active')
approval_status	(character) Approval status (optional)
extras	(list of dataset extra dictionaries) The dataset's extras (optional), extras are arbitrary (key: value) metadata items that can be added to datasets, each extra dictionary should have keys 'key' (a string), 'value' (a string)
packages	(data.frame) The datasets (packages) that belong to the group, a data.frame, where each row has column 'name' (string, the id or name of the dataset) and optionally 'title' (string, the title of the dataset)
groups	(data.frame) The groups to which the dataset belongs (optional), each data.frame row should have one or more of the following columns which identify an existing group: 'id' (the id of the group, string), or 'name' (the name of the group, string), to see which groups exist call group_list()
users	(list of dictionaries) The users that belong to the group, a list of dictionaries each with key 'name' (string, the id or name of the user) and optionally 'capacity' (string, the capacity in which the user is a member of the group)

url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
# Setup
ckanr_setup(url = "https://demo.ckan.org", key = getOption("ckan_demo_key"))

# create a group
(res <- group_create("fruitloops2", description = "A group about fruitloops"))
res$users
res$num_followers

## End(Not run)
```

group_delete	<i>Delete a group</i>
--------------	-----------------------

Description

Delete a group

Usage

```
group_delete(id, url = get_default_url(), key = get_default_key(), ...)
```

Arguments

id	(character) The id of the group. Required.
url	Base url to use. Default: https://data.ontario.ca See also ckanr_setup() and get_default_url()
key	A privileged CKAN API key, Default: your key set with ckanr_setup
...	Curl args passed on to curl::verb-POST (optional)

Examples

```
## Not run:
# Setup
ckanr_setup(url = "https://demo.ckan.org", key = getOption("ckan_demo_key"))

# create a group
(res <- group_create("lions", description = "A group about lions"))

# show the group
group_show(res$id)

# delete the group
group_delete(res)
## or with it's id
# group_delete(res$id)

## End(Not run)
```

group_list

*List groups.***Description**

List groups.

Usage

```
group_list(
  offset = 0,
  limit = 31,
  sort = NULL,
  groups = NULL,
  all_fields = FALSE,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

offset	(numeric) Where to start getting activity items from (optional, default: 0)
limit	(numeric) The maximum number of activities to return (optional, default: 31)
sort	Field to sort on. You can specify ascending (e.g., score desc) or descending (e.g., score asc), sort by two fields (e.g., score desc, price asc), or sort by a function (e.g., sum(x_f, y_f) desc, which sorts by the sum of x_f and y_f in a descending order).

groups	(character) A list of names of the groups to return, if given only groups whose names are in this list will be returned
all_fields	(logical) Return full group dictionaries instead of just names. Default: FALSE
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
group_list(limit = 3)
group_list(limit = 3, as = "json")
group_list(limit = 3, as = "table")

## End(Not run)
```

group_list_authz	<i>List groups the current user can edit</i>
------------------	--

Description

List groups the current user can edit

Usage

```
group_list_authz(
  type = "group",
  groups = NULL,
  all_fields = FALSE,
  include_dataset_count = TRUE,
  include_extras = FALSE,
  include_tags = FALSE,
  include_groups = FALSE,
  include_users = FALSE,
  available_only = FALSE,
  am_member = FALSE,
  offset = NULL,
  limit = NULL,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

type	(character) Group type, defaults to "group".
groups	(character vector) Optional list of group names to filter.
all_fields	(logical) Return full group dictionaries.
include_dataset_count, include_extras, include_tags, include_groups, include_users	(logical) Include additional metadata when all_fields = TRUE.
available_only	(logical) Remove groups already associated to the context dataset.
am_member	(logical) Only return memberships for the current user.
offset	(numeric) Where to start getting activity items from (optional, default: 0)
limit	(numeric) The maximum number of activities to return (optional, default: 31)
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

group_member_create *Make a user a member of a group*

Description

Make a user a member of a group

Usage

```
group_member_create(
  id,
  username,
  role,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

id	(character, ckan_group, or ckan_organization) Identifier for the container.
username	(character or ckan_user) User to add.
role	(character) Desired role ("member", "editor", "admin").
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

group_member_delete	<i>Remove a user from a group</i>
---------------------	-----------------------------------

Description

Remove a user from a group

Usage

```
group_member_delete(
  id,
  username,
  url = get_default_url(),
  key = get_default_key(),
  ...
)
```

Arguments

id	(character, ckan_group, or ckan_organization) Identifier for the container.
username	(character or ckan_user) User to add.
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
...	Curl args passed on to verb-POST (optional)

group_patch	<i>Update a group's metadata</i>
-------------	----------------------------------

Description

Update a group's metadata

Usage

```
group_patch(
  x,
  id,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

x	(list) A list with key-value pairs
id	(character) Resource ID to update (required)
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
# Setup
ckanr_setup(url = "https://demo.ckan.org", key = getOption("ckan_demo_key"))

# Create a package
(res <- group_create("hello-my-world2"))

# Get a resource
grp <- group_show(res$id)
grp$title
grp$author_email

# Make some changes
x <- list(title = "!hello world!", maintainer_email = "hello@world.com")
```

```
group_patch(x, id = grp)

## End(Not run)
```

group_show	<i>Show a package</i>
------------	-----------------------

Description

Show a package

Usage

```
group_show(
  id,
  include_datasets = TRUE,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

id	(character) Package identifier.
include_datasets	(logical) Include a list of the group's datasets. Default: TRUE
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

Details

By default the help and success slots are dropped, and only the result slot is returned. You can request raw json with `as = 'json'` then parse yourself to get the help slot.

Examples

```
## Not run:
res <- group_list()

# via a group name/id
group_show(res[[1]]$name)

# or via an object of class ckan_group
group_show(res[[1]])

# return different data formats
group_show(res[[1]]$name, as = "json")
group_show(res[[1]]$name, as = "table")

## End(Not run)
```

group_update

Update a group

Description

Update a group

Usage

```
group_update(
  x,
  id,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

x	(list) A list with key-value pairs
id	(character) Package identifier
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
# Setup
ckanr_setup(url = "https://demo.ckan.org/", key = getOption("ckan_demo_key"))

# First, create a group
grp <- group_create("water-bears2")
group_show(grp)

## update just chosen things
# Make some changes
x <- list(description = "A group about water bears and people that love them")

# Then update the package
group_update(x, id = grp)

## End(Not run)
```

job_queue

Background job helpers

Description

Manage CKAN's RQ/Redis job queue (requires sysadmin access).

Usage

```
job_list(
  queues = NULL,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)

job_show(
  id,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)

job_clear(
  queues = NULL,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

```

    ...
  )

  job_cancel(
    id,
    url = get_default_url(),
    key = get_default_key(),
    as = "list",
    ...
  )

```

Arguments

queues	(character) Queue names to target.
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)
id	(character) Job identifier.

Examples

```

## Not run:
ckanr_setup(url = "https://demo.ckan.org/", key = "my-ckan-key")
job_list()

## End(Not run)

```

license_list	<i>Return the list of licenses available for datasets on the site.</i>
--------------	--

Description

Return the list of licenses available for datasets on the site.

Usage

```

license_list(
  id,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)

```

Arguments

id	(character) Package identifier.
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
license_list()
license_list(as = "table")
license_list(as = "json")

## End(Not run)
```

member_create

Create or update a membership via member_ endpoints***Description**

Create or update a membership via member_* endpoints

Usage

```
member_create(
  id,
  object,
  object_type,
  capacity,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

id	(character, ckan_group, or ckan_organization) Identifier for the container.
object	(character) The object to add (dataset id, user name, etc.).
object_type	(character) Type of object being added ("user", "package", ...).

capacity	(character) Capacity for the membership.
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to <code>data.frame</code> 's when possible, so the result can vary from a vector, list or <code>data.frame</code> . (required)
...	Curl args passed on to verb-POST (optional)

member_delete	<i>Remove a membership via member_* endpoints</i>
---------------	---

Description

Remove a membership via member_* endpoints

Usage

```
member_delete(
  id,
  object,
  object_type,
  url = get_default_url(),
  key = get_default_key(),
  ...
)
```

Arguments

id	(character, ckan_group, or ckan_organization) Identifier for the container.
object	(character) The object to add (dataset id, user name, etc.).
object_type	(character) Type of object being added ("user", "package", ...).
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
...	Curl args passed on to verb-POST (optional)

member_list	<i>List members for a group or organization</i>
-------------	---

Description

List members for a group or organization

Usage

```
member_list(  
  id,  
  object_type = NULL,  
  capacity = NULL,  
  url = get_default_url(),  
  key = get_default_key(),  
  as = "list",  
  ...  
)
```

Arguments

id	(character, ckan_group, or ckan_organization) Identifier for the container.
object_type	(character) Optional object type filter ("user", "package", etc.).
capacity	(character) Optional capacity filter ("member", "editor", "admin").
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
# ckanr_setup(url = "https://demo.ckan.org/", key = "my-key")  
# member_list("my-group", object_type = "user")
```

member_roles_list	<i>Report available member roles</i>
-------------------	--------------------------------------

Description

Report available member roles

Usage

```
member_roles_list(  
  group_type = "organization",  
  url = get_default_url(),  
  key = get_default_key(),  
  as = "list",  
  ...  
)
```

Arguments

- group_type (character) Either "group" or "organization".
- url Base url to use. Default: <https://demo.ckan.org/> See also [ckanr_setup](#) and [get_default_url](#).
- key A privileged CKAN API key, Default: your key set with [ckanr_setup](#)
- as (character) One of list (default), table, or json. Parsing with table option uses `jsonlite::fromJSON(..., simplifyDataFrame = TRUE)`, which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
- ... Curl args passed on to [verb-POST](#) (optional)

membership	<i>Membership helpers</i>
------------	---------------------------

Description

Utilities for managing CKAN group and organization membership endpoints.

organization_create *Create an organization*

Description

Create an organization

Usage

```
organization_create(
    name = NULL,
    id = NULL,
    title = NULL,
    description = NULL,
    image_url = NULL,
    state = "active",
    approval_status = NULL,
    extras = NULL,
    packages = NULL,
    users = NULL,
    url = get_default_url(),
    key = get_default_key(),
    as = "list",
    ...
)
```

Arguments

name	(character) the name of the organization, a string between 2 and 100 characters long, containing only lowercase alphanumeric characters, • and _
id	the id of the organization (optional)
title	(character) the title of the organization (optional)
description	(character) the description of the organization (optional)
image_url	(character) the URL to an image to be displayed on the organization's page (optional)
state	(character) the current state of the organization, e.g. 'active' or 'deleted', only active organization show up in search results and other lists of organization, this parameter will be ignored if you are not authorized to change the state of the organization (optional). Default: 'active'
approval_status	(character) Approval status
extras	The organization's extras (optional), extras are arbitrary (key: value) metadata items that can be added to organizations, each extra dictionary should have keys 'key' (a string), 'value' (a string) package_relationship_create for the format of relationship dictionaries (optional)

packages	(list of dictionaries) the datasets (packages) that belong to the organization, a list of dictionaries each with keys 'name' (string, the id or name of the dataset) and optionally 'title' (string, the title of the dataset)
users	(character) the users that belong to the organization, a list of dictionaries each with key 'name' (string, the id or name of the user) and optionally 'capacity' (string, the capacity in which the user is a member of the organization)
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to <code>data.frame</code> 's when possible, so the result can vary from a vector, list or <code>data.frame</code> . (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
# Setup
ckanr_setup(url = "https://demo.ckan.org/", key = getOption("ckan_demo_key"))

# create an organization
(res <- organization_create("foobar",
  title = "Foo bars",
  description = "love foo bars"
))
res$name

## End(Not run)
```

organization_delete	<i>Delete an organization</i>
---------------------	-------------------------------

Description

Delete an organization

Usage

```
organization_delete(
  id,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

id	(character) name or id of the organization
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to <code>data.frame</code> 's when possible, so the result can vary from a vector, list or <code>data.frame</code> . (required)
...	Curl args passed on to verb-POST (optional)

Value

an empty list on success

Examples

```
## Not run:
ckanr_setup(url = "https://demo.ckan.org", key = getOption("ckan_demo_key"))

# create an organization
(res <- organization_create("foobar",
  title = "Foo bars",
  description = "love foo bars"
))

# delete the organization just created
res$id
organization_delete(id = res$id)

## End(Not run)
```

organization_list	<i>List organization</i>
-------------------	--------------------------

Description

List organization

Usage

```
organization_list(
  order_by = c("name", "package"),
  decreasing = FALSE,
  organizations = NULL,
  all_fields = TRUE,
  limit = 31,
```

```

    url = get_default_url(),
    key = get_default_key(),
    as = "list",
    ...
)

```

Arguments

<code>order_by</code>	(character, only the first element is used). The field to sort the list by, must be name or packages.
<code>decreasing</code>	(logical). Is the sort-order is decreasing or not.
<code>organizations</code>	(character or NULL). A list of names of the organizations to return. NULL returns all organizations.
<code>all_fields</code>	(logical). Return the name or all fields of the object.
<code>limit</code>	(numeric) The maximum number of organizations to return (optional, default: 31)
<code>url</code>	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
<code>key</code>	A privileged CKAN API key, Default: your key set with ckanr_setup
<code>as</code>	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
<code>...</code>	Curl args passed on to verb-POST (optional)

Examples

```

## Not run:
ckanr_setup(url = "https://demo.ckan.org/")

# list organizations
res <- organization_list()
res[1:2]

# Different data formats
organization_list(as = "json")
organization_list(as = "table")

## End(Not run)

```

organization_list_for_user

List organizations a user can act on

Description

List organizations a user can act on

Usage

```
organization_list_for_user(
  id = NULL,
  permission = "manage_group",
  include_dataset_count = FALSE,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

id	(character or ckan_user) Optional user identifier; defaults to the authenticated user.
permission	(character) Permission to check (defaults to "manage_group").
include_dataset_count	(logical) Include dataset counts in the response.
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

organization_member_create

Make a user a member of an organization

Description

Make a user a member of an organization

Usage

```
organization_member_create(
  id,
  username,
  role,
```

```

url = get_default_url(),
key = get_default_key(),
as = "list",
...
)

```

Arguments

id	(character, ckan_group, or ckan_organization) Identifier for the container.
username	(character or ckan_user) User to add.
role	(character) Desired role ("member", "editor", "admin").
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

organization_member_delete

Remove a user from an organization

Description

Remove a user from an organization

Usage

```

organization_member_delete(
  id,
  username,
  url = get_default_url(),
  key = get_default_key(),
  ...
)

```

Arguments

id	(character, ckan_group, or ckan_organization) Identifier for the container.
username	(character or ckan_user) User to add.
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
...	Curl args passed on to verb-POST (optional)

organization_purge	<i>Purge an organization</i>
--------------------	------------------------------

Description

IMPORTANT: You must be a sysadmin to purge an organization. Once an organization is purged, it is permanently removed from the system.

Usage

```
organization_purge(  
  id,  
  url = get_default_url(),  
  key = get_default_key(),  
  as = "list",  
  ...  
)
```

Arguments

id	(character) name or id of the organization
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to <code>data.frame</code> 's when possible, so the result can vary from a vector, list or <code>data.frame</code> . (required)
...	Curl args passed on to verb-POST (optional)

Value

an empty list on success

Examples

```
## Not run:  
ckanr_setup(url = "https://demo.ckan.org", key = getOption("ckan_demo_key"))  
  
# create an organization  
(res <- organization_create("foobar",  
  title = "Foo bars",  
  description = "love foo bars"  
)  
)  
  
# delete the organization just created  
res$id
```

```

organization_delete(id = res$id)

# purge the organization just deleted
res$id
organization_purge(id = res$id)

## End(Not run)

```

organization_show	<i>Show an organization</i>
-------------------	-----------------------------

Description

Show an organization

Usage

```

organization_show(
  id,
  include_datasets = FALSE,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)

```

Arguments

id	(character) Organization id or name.
include_datasets	(logical). Whether to include a list of the organization datasets
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

Details

By default the help and success slots are dropped, and only the result slot is returned. You can request raw json with `as = 'json'` then parse yourself to get the help slot.

Examples

```
## Not run:
ckanr_setup(url = "https://demo.ckan.org/", key = getOption("ckan_demo_key"))

res <- organization_create("stuffthings2")
organization_show(res$id)

## End(Not run)
```

`package_activity_list` *Return a list of the package's activity*

Description

Return a list of the package's activity

Usage

```
package_activity_list(
  id,
  offset = 0,
  limit = 31,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

<code>id</code>	(character) Package identifier.
<code>offset</code>	(numeric) Where to start getting activity items from (optional, default: 0)
<code>limit</code>	(numeric) The maximum number of activities to return (optional, default: 31)
<code>url</code>	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
<code>key</code>	A privileged CKAN API key, Default: your key set with ckanr_setup
<code>as</code>	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
<code>...</code>	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
# Setup
ckanr_setup(url = "https://demo.ckan.org/", key = getOption("ckan_demo_key"))

# create a package
(res <- package_create("owls64"))

# list package activity
package_activity_list(res$id)

# make a change
x <- list(maintainer = "Jane Forest")
package_update(x, res)

# list activity again
package_activity_list(res)

# output different data formats
package_activity_list(res$id, as = "table")
package_activity_list(res$id, as = "json")

## End(Not run)
```

package_collaborator_create
Create or update a dataset collaborator

Description

Create or update a dataset collaborator

Usage

```
package_collaborator_create(
  id,
  user_id,
  capacity,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

id (character or ckan_package) Dataset identifier or object.
user_id (character or ckan_user) User identifier or object.

capacity	(character) Collaborator role ("member", "editor", "admin").
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to <code>data.frame</code> 's when possible, so the result can vary from a vector, list or <code>data.frame</code> . (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
ckanr_setup(url = "https://demo.ckan.org/", key = "my-key")
package_collaborator_create("my-dataset", "new-user", capacity = "editor")

## End(Not run)
```

package_collaborator_delete

Delete a dataset collaborator

Description

Delete a dataset collaborator

Usage

```
package_collaborator_delete(
  id,
  user_id,
  url = get_default_url(),
  key = get_default_key(),
  ...
)
```

Arguments

id	(character or ckan_package) Dataset identifier or object.
user_id	(character or ckan_user) User identifier or object.
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
ckanr_setup(url = "https://demo.ckan.org/", key = "my-key")
package_collaborator_delete("my-dataset", "new-user")

## End(Not run)
```

package_collaborator_list

List collaborators for a dataset

Description

List collaborators for a dataset

Usage

```
package_collaborator_list(
  id,
  capacity = NULL,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

id	(character or ckan_package) Dataset identifier or object.
capacity	(character) Optional capacity filter ("member", "editor", "admin").
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
ckanr_setup(url = "https://demo.ckan.org/", key = "my-key")
package_collaborator_list("my-dataset")

## End(Not run)
```

package_collaborator_list_for_user

List packages a user collaborates on

Description

List packages a user collaborates on

Usage

```
package_collaborator_list_for_user(
  id,
  capacity = NULL,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

id	(character or ckan_user) User identifier or object.
capacity	(character) Optional capacity filter.
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
ckanr_setup(url = "https://demo.ckan.org/", key = "my-key")
package_collaborator_list_for_user("sckotttie")

## End(Not run)
```

package_collaborators *Package collaborator utilities*

Description

Package collaborator utilities

package_create	<i>Create a package</i>
----------------	-------------------------

Description

Create a package

Usage

```
package_create(
  name = NULL,
  title = NULL,
  private = FALSE,
  author = NULL,
  author_email = NULL,
  maintainer = NULL,
  maintainer_email = NULL,
  license_id = NULL,
  notes = NULL,
  package_url = NULL,
  version = NULL,
  state = "active",
  type = NULL,
  resources = NULL,
  tags = NULL,
  extras = NULL,
  relationships_as_object = NULL,
  relationships_as_subject = NULL,
  groups = NULL,
  owner_org = NULL,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

name	(character) the name of the new dataset, must be between 2 and 100 characters long and contain only lowercase alphanumeric characters, - and _, e.g. 'warand-peace'
title	(character) the title of the dataset (optional, default: same as name)
private	(logical) whether the dataset should be private (optional, default: FALSE), requires a value for owner_org if TRUE
author	(character) the name of the dataset's author (optional)
author_email	(character) the email address of the dataset's author (optional)

<code>maintainer</code>	(character) the name of the dataset's maintainer (optional)
<code>maintainer_email</code>	(character) the email address of the dataset's maintainer (optional)
<code>license_id</code>	(license id string) - the id of the dataset's license, see <code>license_list()</code> for available values (optional)
<code>notes</code>	(character) a description of the dataset (optional)
<code>package_url</code>	(character) a URL for the dataset's source (optional)
<code>version</code>	(string, no longer than 100 characters) - (optional)
<code>state</code>	(character) the current state of the dataset, e.g. 'active' or 'deleted', only active datasets show up in search results and other lists of datasets, this parameter will be ignored if you are not authorized to change the state of the dataset (optional, default: 'active')
<code>type</code>	(character) the type of the dataset (optional), IDatasetForm plugins associate themselves with different dataset types and provide custom dataset handling behaviour for these types
<code>resources</code>	(list of resource dictionaries) - the dataset's resources, see <code>resource_create()</code> for the format of resource dictionaries (optional)
<code>tags</code>	(list of tag dictionaries) - the dataset's tags, see <code>tag_create()</code> for the format of tag dictionaries (optional)
<code>extras</code>	(list of dataset extra dictionaries) - the dataset's extras (optional), extras are arbitrary (key: value) metadata items that can be added to datasets, each extra dictionary should have keys 'key' (a string), 'value' (a string)
<code>relationships_as_object</code>	(list of relationship dictionaries) - see <code>package_relationship_create</code> for the format of relationship dictionaries (optional)
<code>relationships_as_subject</code>	(list of relationship dictionaries) - see <code>package_relationship_create</code> for the format of relationship dictionaries (optional)
<code>groups</code>	(data.frame) the groups to which the dataset belongs, each row should have one or more of the following columns which identify an existing group: 'id' (the id of the group, string), or 'name' (the name of the group, string), to see which groups exist call <code>group_list()</code> . see example (optional)
<code>owner_org</code>	(character) the id of the dataset's owning organization, see <code>organization_list()</code> or <code>organization_list_for_user</code> for available values (optional)
<code>url</code>	Base url to use. Default: <code>https://demo.ckan.org/</code> See also <code>ckanr_setup</code> and <code>get_default_url</code> .
<code>key</code>	A privileged CKAN API key, Default: your key set with <code>ckanr_setup</code>
<code>as</code>	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
<code>...</code>	Curl args passed on to <code>verb-POST</code> (optional)

Examples

```
## Not run:
# Setup
ckanr_setup(url = "https://demo.ckan.org/", key = getOption("ckan_demo_key"))

# create a package
## Example 1
(res <- package_create("foobar4", author = "Jane Doe"))
res$author

## Example 2 - create package, add a resource
(res <- package_create("helloworld", author = "Jane Doe"))

# include a group
# package_create("brownbear", groups = data.frame(id = "some-id"))

## End(Not run)
```

package_create_default_resource_views

Create default views for all resources in a dataset

Description

Create default views for all resources in a dataset

Usage

```
package_create_default_resource_views(
  package,
  create_datastore_views = FALSE,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

package	(character or ckan_package) Dataset identifier.
create_datastore_views	(logical) When TRUE, only create views that require DataStore-backed resources (used when DataPusher finishes ingesting data).
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup

as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to <code>data.frame</code> 's when possible, so the result can vary from a vector, list or <code>data.frame</code> . (required)
...	Curl args passed on to verb-POST (optional)

package_dataset_extras

Dataset maintenance helpers

Description

Additional CKAN 2.11 dataset endpoints for bulk edits and destructive actions.

package_delete

Delete a package

Description

Delete a package

Usage

```
package_delete(id, url = get_default_url(), key = get_default_key(), ...)
```

Arguments

id	(character) The id of the package. Required.
url	Base url to use. Default: <code>https://data.ontario.ca</code> See also ckanr_setup() and get_default_url()
key	A privileged CKAN API key, Default: your key set with ckanr_setup
...	Curl args passed on to curl::verb-POST (optional)

Examples

```
## Not run:
# Setup
ckanr_setup(url = "https://demo.ckan.org", key = getOption("ckan_demo_key"))

# create a package
(res <- package_create("lions-bears-tigers"))

# show the package
package_show(res)
```

```
# delete the package
package_delete(res)

## End(Not run)
```

package_list	<i>List datasets.</i>
--------------	-----------------------

Description

List datasets.

Usage

```
package_list(
  offset = 0,
  limit = 31,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

offset	(numeric) Where to start getting activity items from (optional, default: 0)
limit	(numeric) The maximum number of activities to return (optional, default: 31)
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
package_list()
package_list(as = "json")
package_list(as = "table")

package_list(url = "https://data.nhm.ac.uk")

## End(Not run)
```

package_list_current *List current packages with resources.*

Description

List current packages with resources.

Usage

```
package_list_current(  
  offset = 0,  
  limit = 31,  
  url = get_default_url(),  
  key = get_default_key(),  
  as = "list",  
  ...  
)
```

Arguments

offset	(numeric) Where to start getting activity items from (optional, default: 0)
limit	(numeric) The maximum number of activities to return (optional, default: 31)
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:  
package_list_current()  
package_list_current(as = "json")  
package_list_current(as = "table")  
  
## End(Not run)
```

package_owner_org_update

Move a dataset to another organization

Description

Move a dataset to another organization

Usage

```
package_owner_org_update(  
  id,  
  organization_id,  
  url = get_default_url(),  
  key = get_default_key(),  
  ...  
)
```

Arguments

id	(character or ckan_package) Dataset identifier.
organization_id	(character or ckan_organization) Owning organization identifier.
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:  
package_owner_org_update("dataset-id", organization_id = "target-org")  
  
## End(Not run)
```

package_patch

Update a package's metadata

Description

Update a package's metadata

Usage

```
package_patch(
  x,
  id = NULL,
  extras = NULL,
  http_method = "GET",
  key = get_default_key(),
  url = get_default_url(),
  as = "list",
  ...
)
```

Arguments

x	(list) A list with key-value pairs
id	(character) Resource ID to update (optional, required if x does not have an "id" field)
extras	(character vector) - the dataset's extras (optional), extras are arbitrary (key: value) metadata items that can be added to datasets, each extra dictionary should have keys 'key' (a string), 'value' (a string)
http_method	(character) which HTTP method (verb) to use; one of "GET" or "POST". Default: "GET"
key	A privileged CKAN API key, Default: your key set with ckanr_setup
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
# Setup
ckanr_setup(url = "https://demo.ckan.org", key = getOption("ckan_demo_key"))

# Create a package
(res <- package_create("hello-world13", author = "Jane Doe"))

# Get a resource
res <- package_show(res$id)
res$title

# patch
package_patch(res, extras = list(list(key = "foo", value = "bar")))
unclass(package_show(res))
```

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

```
package_relationship_create
      Create a dataset relationship
```

Description

Create a dataset relationship

Usage

```
package_relationship_create(
  subject,
  object,
  relationship_type,
  comment = NULL,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

subject	(character or ckan_package) Dataset acting as the subject in the relationship.
object	(character or ckan_package) Dataset acting as the object in the relationship.
relationship_type	(character) Relationship type ("depends_on", "derives_from", etc.).
comment	(character) Optional note attached to the relationship.
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
package_relationship_create("dataset-a", "dataset-b", relationship_type = "derives_from")

## End(Not run)
```

package_relationship_delete
Delete a dataset relationship

Description

Delete a dataset relationship

Usage

```
package_relationship_delete(  
  subject,  
  object,  
  relationship_type,  
  url = get_default_url(),  
  key = get_default_key(),  
  ...  
)
```

Arguments

subject	(character or ckan_package) Dataset acting as the subject in the relationship.
object	(character or ckan_package) Dataset acting as the object in the relationship.
relationship_type	(character) Relationship type ("depends_on", "derives_from", etc.).
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
...	Curl args passed on to verb-POST (optional)

package_relationship_update
Update an existing dataset relationship

Description

Update an existing dataset relationship

Usage

```
package_relationship_update(
  subject,
  object,
  relationship_type,
  comment = NULL,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

subject	(character or ckan_package) Dataset acting as the subject in the relationship.
object	(character or ckan_package) Dataset acting as the object in the relationship.
relationship_type	(character) Relationship type ("depends_on", "derives_from", etc.).
comment	(character) Optional note attached to the relationship.
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

package_relationships *Package relationship helpers*

Description

Utilities for managing dataset relationships via CKAN's `package_relationship_*` endpoints.

`package_relationships_list`*List relationships for a dataset*

Description

List relationships for a dataset

Usage

```
package_relationships_list(  
  id,  
  id2 = NULL,  
  relationship_type = NULL,  
  url = get_default_url(),  
  key = get_default_key(),  
  as = "list",  
  ...  
)
```

Arguments

<code>id</code>	(character or <code>ckan_package</code>) The dataset to inspect.
<code>id2</code>	(character or <code>ckan_package</code>) Optional secondary dataset identifier to scope results.
<code>relationship_type</code>	(character) Filter on relationship type (see package_relationship_create()).
<code>url</code>	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
<code>key</code>	A privileged CKAN API key, Default: your key set with ckanr_setup
<code>as</code>	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to <code>data.frame</code> 's when possible, so the result can vary from a vector, list or <code>data.frame</code> . (required)
<code>...</code>	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:  
ckanr_setup(url = "https://demo.ckan.org/", key = "my-key")  
package_relationships_list("my-dataset")  
  
## End(Not run)
```

`package_resource_reorder`*Reorder resources for a dataset*

Description

Reorder resources for a dataset

Usage

```
package_resource_reorder(  
  id,  
  order,  
  url = get_default_url(),  
  key = get_default_key(),  
  as = "list",  
  ...  
)
```

Arguments

<code>id</code>	(character or <code>ckan_package</code>) Dataset identifier.
<code>order</code>	(character vector) Resource IDs in the desired order.
<code>url</code>	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
<code>key</code>	A privileged CKAN API key, Default: your key set with ckanr_setup
<code>as</code>	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to <code>data.frame</code> 's when possible, so the result can vary from a vector, list or <code>data.frame</code> . (required)
<code>...</code>	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:  
pkg <- package_show("my-dataset")  
package_resource_reorder(pkg,  
  order = rev(vapply(pkg$resources, `[`, "id", FUN.VALUE = character(1)))  
)  
  
## End(Not run)
```

package_revise	<i>Revise a dataset using match/filter/update semantics</i>
----------------	---

Description

Revise a dataset using match/filter/update semantics

Usage

```
package_revise(
  match = NULL,
  filter = NULL,
  update = NULL,
  include = NULL,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

match	(list) Key/value pairs identifying the dataset to revise. Required unless using flattened keys.
filter	(character or list) Patterns describing fields to remove before the update runs.
update	(list) Values to set after filtering. Supports flattened keys.
include	(character or list) Optional patterns delimiting which fields are returned in the response.
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to <code>data.frame</code> 's when possible, so the result can vary from a vector, list or <code>data.frame</code> . (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
package_revise(
  match = list(name = "source-dataset"),
  update = list(notes = "New description")
)

## End(Not run)
```

`package_revision_list` *Return a dataset (package's) revisions as a list of dictionaries.*

Description

Return a dataset (package's) revisions as a list of dictionaries.

Usage

```
package_revision_list(
  id,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

<code>id</code>	(character) Package identifier.
<code>url</code>	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
<code>key</code>	A privileged CKAN API key, Default: your key set with ckanr_setup
<code>as</code>	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
<code>...</code>	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
# Setup
ckanr_setup(url = "https://demo.ckan.org/", key = getOption("ckan_demo_key"))

# create a package
(res <- package_create("dolphins"))

# list package revisions
package_revision_list(res$id)

# Make change to the package
x <- list(title = "dolphins and things")
package_patch(x, id = res$id)

# list package revisions
package_revision_list(res$id)
```

```
# Output different formats
package_revision_list(res$id, as = "table")
package_revision_list(res$id, as = "json")

## End(Not run)
```

package_search	<i>Search for packages.</i>
----------------	-----------------------------

Description

Search for packages.

Usage

```
package_search(
  q = "*:*",
  fq = NULL,
  sort = NULL,
  rows = NULL,
  start = NULL,
  facet = FALSE,
  facet.limit = NULL,
  facet.field = NULL,
  facet.mincount = NULL,
  include_drafts = FALSE,
  include_private = FALSE,
  use_default_schema = FALSE,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

q	Query terms, defaults to ':', or everything.
fq	Filter query, this does not affect the search, only what gets returned
sort	Field to sort on. You can specify ascending (e.g., score desc) or descending (e.g., score asc), sort by two fields (e.g., score desc, price asc), or sort by a function (e.g., sum(x_f, y_f) desc, which sorts by the sum of x_f and y_f in a descending order).
rows	Number of records to return. Defaults to 10.
start	Record to start at, default to beginning.
facet	(logical) Whether to return facet results or not. Default: FALSE

facet.limit	(numeric) This param indicates the maximum number of constraint counts that should be returned for the facet fields. A negative value means unlimited. Default: 100. Can be specified on a per field basis.
facet.field	(character) This param allows you to specify a field which should be treated as a facet. It will iterate over each Term in the field and generate a facet count using that Term as the constraint. This parameter can be specified multiple times to indicate multiple facet fields. None of the other params in this section will have any effect without specifying at least one field name using this param.
facet.mincount	(integer) the minimum counts for facet fields should be included in the results
include_drafts	(logical) if TRUE draft datasets will be included. A user will only be returned their own draft datasets, and a sysadmin will be returned all draft datasets. default: FALSE. first CKAN version: 2.6.1; dropped from request if CKAN version is older or if CKAN version isn't available via ckan_version()
include_private	(logical) if TRUE private datasets will be included. Only private datasets from the user's organizations will be returned and sysadmins will be returned all private datasets. default: FALSE first CKAN version: 2.6.1; dropped from request if CKAN version is older or if CKAN version isn't available via ckan_version()
use_default_schema	(logical) use default package schema instead of a custom schema defined with an IDatasetForm plugin. default: FALSE first CKAN version: 2.3.5; dropped from request if CKAN version is older or if CKAN version isn't available via ckan_version()
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
ckanr_setup(url = "https://demo.ckan.org", key = getOption("ckan_demo_key"))

package_search(q = "*:*")
package_search(q = "*:*", rows = 2, as = "json")
package_search(q = "*:*", rows = 2, as = "table")

package_search(q = "*:*", sort = "score asc")
package_search(q = "*:*", fq = "num_tags:[3 TO *]")$count
package_search(q = "*:*", fq = "num_tags:[2 TO *]")$count
package_search(q = "*:*", fq = "num_tags:[1 TO *]")$count

## End(Not run)
```

package_show	<i>Show a package.</i>
--------------	------------------------

Description

Show a package.

Usage

```
package_show(
  id,
  use_default_schema = FALSE,
  http_method = "GET",
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

id	(character/ckan_package) Package identifier, or a ckan_package object
use_default_schema	(logical) Use default package schema instead of a custom schema defined with an IDatasetForm plugin. Default: FALSE
http_method	(character) which HTTP method (verb) to use; one of "GET" or "POST". Default: "GET"
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

Details

By default the help and success slots are dropped, and only the result slot is returned. You can request raw json with `as = 'json'` then parse yourself to get the help slot.

Examples

```
## Not run:
# Setup
ckanr_setup(
  url = "https://demo.ckan.org/",
  key = getOption("ckan_demo_key")
)

# create a package
(res <- package_create("purposeful55"))

# show package
## From the output of package_create
package_show(res)
## Or, from the ID
package_show(res$id)

# get data back in different formats
package_show(res$id, as = "json")
package_show(res$id, as = "table")

# use default schema or not
package_show(res$id, TRUE)

## End(Not run)
```

package_update	<i>Update a package</i>
----------------	-------------------------

Description

This function updates all package metadata fields. Any update will also set the metadata key "last_updated". Any omitted metadata fields will be overwritten.

Usage

```
package_update(
  x,
  id,
  http_method = "GET",
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

x	(list) A list with key-value pairs
id	(character) Package identifier
http_method	(character) which HTTP method (verb) to use; one of "GET" or "POST". Default: "GET"
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to <code>data.frame</code> 's when possible, so the result can vary from a vector, list or <code>data.frame</code> . (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
# Setup
ckanr_setup(url = "https://demo.ckan.org/", key = getOption("ckan_demo_key"))

# Step 1: get the dataset details as R list
ds_id <- "my-dataset-id-md5-hash"
ds <- ckanr::package_show(ds_id, as = "table")

# Step 2: update selected fields
ds$title <- "An updated title"
ds$description <- "Only title and description have been updated."
# ds contains all other package data, including tags and resources

# Step 3a: Update the dataset on CKAN with locally modified metadata `ds`
result <- ckanr::package_update(ds, ds_id)
# Replace existing package metadata

# Step 3b: Possible or intended data loss
# Any metadata fields missing from `ds` will be deleted in the package
del(ds$description)
result_with_deleted_description <- ckanr::package_update(ds, ds_id)

## End(Not run)
```

ping

Ping a CKAN server to test that it's up or down.

Description

Ping a CKAN server to test that it's up or down.

Usage

```
ping(url = get_default_url(), key = get_default_key(), as = "logical", ...)
```

Arguments

url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to <code>data.frame</code> 's when possible, so the result can vary from a vector, list or <code>data.frame</code> . (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
ping()
ping(as = "json")

## End(Not run)
```

related_create

Create a related item

Description

Create a related item

Usage

```
related_create(
  id,
  title,
  type,
  description = NULL,
  related_id = NULL,
  related_url = NULL,
  image_url = NULL,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

id	(character) id of package that the related item should be added to. This should be an alphanumeric string. Required.
title	(character) Title of the related item. Required.
type	(character) The type of the related item. One of API, application, idea, news article, paper, post or visualization. Required.
description	(character) description (optional). Optional
related_id	(character) An id to assign to the related item. If blank, an ID will be assigned for you. Optional
related_url	(character) A url to associated with the related item. Optional
image_url	(character) A url to associated image. Optional
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to <code>data.frame</code> 's when possible, so the result can vary from a vector, list or <code>data.frame</code> . (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
# Setup
ckanr_setup(url = "https://demo.ckan.org/", key = getOption("ckan_demo_key"))

# create a package
(res <- package_create("hello-mars"))

# create a related item
related_create(res, title = "asfdaf", type = "idea")

# pipe operations together
package_create("foobbbbbbarrrr") %>%
  related_create(
    title = "my resource",
    type = "visualization"
  )

## End(Not run)
```

related_delete	<i>Delete a related item.</i>
----------------	-------------------------------

Description

Delete a related item.

Usage

```
related_delete(id, url = get_default_url(), key = get_default_key(), ...)
```

Arguments

id	(character) Resource identifier.
url	Base url to use. Default: https://data.ontario.ca See also ckanr_setup() and get_default_url() .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
...	Curl args passed on to crul::verb-POST (optional)

Examples

```
## Not run:
# Setup
ckanr_setup(url = "https://demo.ckan.org/", key = getOption("ckan_demo_key"))

# create a package and a related item
res <- package_create("hello-venus2") %>%
  related_create(
    title = "my resource",
    type = "visualization"
  )

# show the related item
related_delete(res)
## or with id itself:
## related_delete(res$id)

## End(Not run)
```

related_list	<i>List related items</i>
--------------	---------------------------

Description

List related items

Usage

```
related_list(  
  offset = 0,  
  limit = 31,  
  url = get_default_url(),  
  key = get_default_key(),  
  as = "list",  
  ...  
)
```

Arguments

offset	(numeric) Where to start getting activity items from (optional, default: 0)
limit	(numeric) The maximum number of activities to return (optional, default: 31)
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:  
related_list()  
related_list(as = "json")  
related_list(as = "table")  
  
## End(Not run)
```

related_show	<i>Show a related item</i>
--------------	----------------------------

Description

Show a related item

Usage

```
related_show(
  id,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

id	(character) Related item identifier.
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

Details

By default the help and success slots are dropped, and only the result slot is returned. You can request raw json with `as = 'json'` then parse yourself to get the help slot.

Examples

```
## Not run:
# Setup
ckanr_setup(url = "https://demo.ckan.org/", key = getOption("ckan_demo_key"))

# create a package and a related item
res <- package_create("hello-pluto2") %>%
  related_create(
    title = "my resource",
    type = "visualization"
  )
```

```

# show the related item
related_show(res)
related_show(res$id)

# get data back in different formats
related_show(res, as = "json")
related_show(res, as = "table")

## End(Not run)

```

resource_create	<i>Create a resource</i>
-----------------	--------------------------

Description

Create a resource

Usage

```

resource_create(
  package_id = NULL,
  rcurl = NULL,
  revision_id = NULL,
  description = NULL,
  format = NULL,
  hash = NULL,
  name = NULL,
  resource_type = NULL,
  mimetype = NULL,
  mimetype_inner = NULL,
  webstore_url = NULL,
  cache_url = NULL,
  size = NULL,
  created = NULL,
  last_modified = NULL,
  cache_last_updated = NULL,
  webstore_last_updated = NULL,
  upload = NULL,
  extras = NULL,
  http_method = "GET",
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)

```

Arguments

package_id	(character) id of package that the resource should be added to. This should be an alphanumeric string. Required.
rcurl	(character) url of resource. Required.
revision_id	(character) revision id (optional)
description	(character) description (optional). Required.
format	(character) format (optional)
hash	(character) hash (optional)
name	(character) name (optional). Required.
resource_type	(character) resource type (optional)
mimetype	(character) mime type (optional)
mimetype_inner	(character) mime type inner (optional)
webstore_url	(character) webstore url (optional)
cache_url	(character) cache url(optional)
size	(integer) size (optional)
created	(character) iso date string (optional)
last_modified	(character) iso date string (optional)
cache_last_updated	(character) iso date string (optional)
webstore_last_updated	(character) iso date string (optional)
upload	(character) A path to a local file (optional)
extras	(list) - the resources' extra metadata fields (optional)
http_method	(character) which HTTP method (verb) to use; one of "GET" or "POST". Default: "GET"
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
# Setup
ckanr_setup(
  url = "https://demo.ckan.org/",
  key = getOption("ckan_demo_key")
)
```

```

# create a package
(res <- package_create("foobarrrr", author = "Jane Doe"))

# then create a resource
file <- system.file("examples", "actinidiaceae.csv", package = "ckanr")
(xx <- resource_create(
  package_id = res$id,
  description = "my resource",
  name = "bears",
  upload = file,
  extras = list(species = "grizzly"),
  rcurl = "http://google.com"
))

package_create("foobbbbbarrrr") %>%
  resource_create(
    description = "my resource",
    name = "bearsareus",
    upload = file,
    extras = list(my_extra = "some value"),
    rcurl = "http://google.com"
  )

## End(Not run)

```

resource_create_default_resource_views

Create default views for a single resource

Description

Create default views for a single resource

Usage

```

resource_create_default_resource_views(
  resource,
  package = NULL,
  create_datastore_views = FALSE,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)

```

Arguments

resource (character or ckan_resource) Resource identifier or object.

package	(optional) Dataset identifier or ckan_package object.
create_datastore_views	(logical) When TRUE, only create views that require DataStore-backed resources (used when DataPusher finishes ingesting data).
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

resource_delete	<i>Delete a resource.</i>
-----------------	---------------------------

Description

Delete a resource.

Usage

```
resource_delete(id, url = get_default_url(), key = get_default_key(), ...)
```

Arguments

id	(character) Resource identifier.
url	Base url to use. Default: https://data.ontario.ca See also ckanr_setup() and get_default_url() .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
...	Curl args passed on to crul::verb-POST (optional)

Examples

```
## Not run:
# Setup
ckanr_setup(url = "https://demo.ckan.org/", key = Sys.getenv("CKAN_DEMO_KEY"))

# create a package
(res <- package_create("yellow9"))

# then create a resource
file <- system.file("examples", "actinidiaceae.csv", package = "ckanr")
(xx <- resource_create(res,
  description = "my resource",
  name = "bears",
```

```

    upload = file,
    rcurl = "http://google.com"
  ))

  # delete the resource
  resource_delete(xx)

  ## End(Not run)

```

resource_patch	<i>Update a resource's metadata</i>
----------------	-------------------------------------

Description

Update a resource's metadata

Usage

```

resource_patch(
  x,
  id,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)

```

Arguments

x	(list) A list with key-value pairs
id	(character) Resource ID to update (required)
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

Examples

```

## Not run:
# Setup
ckanr_setup(url = "https://demo.ckan.org", key = getOption("ckan_demo_key"))

# create a package

```

```
(res <- package_create("twist", author = "Alexandria"))

# then create a resource
file <- system.file("examples", "actinidiaceae.csv", package = "ckanr")
(xx <- resource_create(package_id = res$id, description = "my resource"))

# Get a resource
res <- resource_show(xx$id)
res$description

# Make some changes
x <- list(description = "My newer description")
z <- resource_patch(x, id = res)
z$description

# Add an extra key:value pair
extra <- list("extra_key" = "my special value")
zz <- resource_patch(extra, id = res)
zz$extra_key

## End(Not run)
```

resource_search

Search for resources.

Description

Search for resources.

Usage

```
resource_search(
  q,
  sort = NULL,
  offset = NULL,
  limit = NULL,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

q	Query terms. It is a string of the form <code>field:term</code> or a vector/list of strings, each of the same form. Within each string, <code>field</code> is a field or extra field on the Resource domain object. If <code>field</code> is <code>hash</code> , then an attempt is made to match the term as a <i>prefix</i> of the <code>Resource.hash</code> field. If <code>field</code> is an extra field, then an attempt is made to match against the extra fields stored against the Resource.
---	--

sort	Field to sort on. You can specify ascending (e.g., score desc) or descending (e.g., score asc), sort by two fields (e.g., score desc, price asc), or sort by a function (e.g., sum(x_f, y_f) desc, which sorts by the sum of x_f and y_f in a descending order).
offset	Record to start at, default to beginning.
limit	Number of records to return.
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
resource_search(q = "name:data")
resource_search(q = "name:data", as = "json")
resource_search(q = "name:data", as = "table")
resource_search(q = "name:data", limit = 2, as = "table")
resource_search(q = c("description:encoded", "name:No.2"), url = "demo.ckan.org")

## End(Not run)
```

resource_show	<i>Show a resource.</i>
---------------	-------------------------

Description

Show a resource.

Usage

```
resource_show(
  id,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

id	(character) Resource identifier.
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to <code>data.frame</code> 's when possible, so the result can vary from a vector, list or <code>data.frame</code> . (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
# Setup
ckanr_setup(
  url = "https://demo.ckan.org/",
  key = Sys.getenv("CKAN_DEMO_KEY")
)

# create a package
(res <- package_create("yellow7"))

# then create a resource
file <- system.file("examples", "actinidiaceae.csv", package = "ckanr")
(xx <- resource_create(
  package_id = res$id,
  description = "my resource",
  name = "bears",
  upload = file,
  rcurl = "http://google.com"
))

# show the resource
resource_show(xx$id)

# eg. from the NHM CKAN store
resource_show(
  id = "05ff2255-c38a-40c9-b657-4ccb55ab2feb",
  url = "http://data.nhm.ac.uk"
)

## End(Not run)
```

resource_update	<i>Update a resource</i>
-----------------	--------------------------

Description

This function can be used to update a resource's file attachment and "extra" metadata fields. Any update will also set the metadata key "last_updated". Any omitted metadata fields will be overwritten.

To update selected metadata fields and retain all others unchanged, it is highly recommended to first retrieve the full resource metadata using `resource_show`, then update these metadata as required, then update the given resource with `resource_update` using the locally updated metadata.

If a resource file is updated, then the new file must exist on a local path. R objects cannot directly be used to update a resource file; instead, they have to be written to a file, e.g. using `tempfile()` - see example.

For convenience, CKAN base url and API key default to the global options, which are set by `ckanr_setup`.

Usage

```
resource_update(
  id,
  path = NULL,
  extras = list(),
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

<code>id</code>	(character) Resource ID to update (required)
<code>path</code>	(character) Local path of the file to upload (optional)
<code>extras</code>	(list) - the resources' extra metadata fields (optional)
<code>url</code>	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
<code>key</code>	A privileged CKAN API key, Default: your key set with ckanr_setup
<code>as</code>	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
<code>...</code>	Curl args passed on to verb-POST (optional)

Value

The HTTP response from CKAN, formatted as list (default), table, or JSON.

References

http://docs.ckan.org/en/latest/api/index.html#ckan.logic.action.create.resource_create

Examples

```
## Not run:
ckanr_setup(url = "https://demo.ckan.org/", key = getOption("ckan_demo_key"))

# Get file
path <- system.file("examples", "actinidiaceae.csv", package = "ckanr")

# Create package, then a resource within that package
(res <- package_create("newpackage10"))
(xx <- resource_create(
  package_id = res$id,
  description = "my resource",
  name = "bears",
  upload = path,
  rcurl = "http://google.com"
))

# Modify dataset, here lowercase strings in one column
dat <- read.csv(path, stringsAsFactors = FALSE)
dat$Family <- tolower(dat$Family)
newpath <- tempfile(fileext = ".csv")
write.csv(dat, file = newpath, row.names = FALSE)

# Upload modified dataset
## Directly from output of resource_create
resource_update(xx, path = newpath)

## or from the resource id
resource_update(xx$id, path = newpath)

## optionally include extra tags
resource_update(xx$id,
  path = newpath,
  extras = list(some = "metadata")
)

# Update a resource's extra tags
## add extra tags without uploading a new file
resource_update(id,
  extras = list(some = "metadata")
)

## or remove all extra tags
resource_update(id, extras = list())
```

```
#####
# Using default settings
ckanr_setup(url = "http://demo.ckan.org/", key = "my-demo-ckan-org-api-key")
path <- system.file("examples", "actinidiaceae.csv", package = "ckanr")
resource_update(id = "an-existing-resource-id", path = path)

# Using an R object written to a tempfile, and implicit CKAN URL and API key
write.csv(data <- installed.packages(), path <- tempfile(fileext = ".csv"))
ckanr_setup(url = "http://demo.ckan.org/", key = "my-demo-ckan-org-api-key")
resource_update(id = "an-existing-resource-id", path = path)

# Testing: see ?ckanr_setup to set default test CKAN url, key, package id
ckanr_setup(
  test_url = "http://my-ckan.org/",
  test_key = "my-ckan-api-key",
  test_did = "an-existing-package-id",
  test_rid = "an-existing-resource-id"
)
resource_update(
  id = get_test_rid(),
  path = system.file("examples",
    "actinidiaceae.csv",
    package = "ckanr"
  ),
  key = get_test_key(),
  url = get_test_url()
)

# other file formats
## html
path <- system.file("examples", "mapbox.html", package = "ckanr")

# Create package, then a resource within that package
(res <- package_create("mappkg"))
(xx <- resource_create(
  package_id = res$id,
  description = "a map, yay",
  name = "mapyay",
  upload = path,
  rcurl = "http://google.com"
))
browseURL(xx$url)

# Modify dataset, here lowercase strings in one column
dat <- readLines(path)
dat <- sub("-111.06", "-115.06", dat)
newpath <- tempfile(fileext = ".html")
cat(dat, file = newpath, sep = "\n")

# Upload modified dataset
## Directly from output of resource_create
(xxx <- resource_update(xx, path = newpath))
```

```
browseURL(xxx$url)

## End(Not run)
```

resource_view_clear	<i>Clear resource views</i>
---------------------	-----------------------------

Description

Clear resource views

Usage

```
resource_view_clear(
  view_types = NULL,
  url = get_default_url(),
  key = get_default_key(),
  ...
)
```

Arguments

view_types	(character vector) Optional subset of view types to delete. When NULL, all views are removed.
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
...	Curl args passed on to verb-POST (optional)

resource_view_create	<i>Create a resource view</i>
----------------------	-------------------------------

Description

Create a resource view

Usage

```
resource_view_create(
  resource,
  view_type,
  title,
  description = NULL,
  config = NULL,
  filter_fields = NULL,
```

```

    filter_values = NULL,
    url = get_default_url(),
    key = get_default_key(),
    as = "list",
    ...
)

```

Arguments

resource	(character or ckan_resource) Parent resource identifier.
view_type	(character) Plugin type registered on the CKAN site.
title	(character) Title assigned to the view.
description	(character) Optional description.
config	(list) Arbitrary configuration list passed to the plugin.
filter_fields, filter_values	Optional filter parameters for filterable views.
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

resource_view_delete *Delete a resource view*

Description

Delete a resource view

Usage

```
resource_view_delete(id, url = get_default_url(), key = get_default_key(), ...)
```

Arguments

id	(character or ckan_resource_view) View identifier.
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
...	Curl args passed on to verb-POST (optional)

Value

(bool) The result of the deletion, TRUE if the resource view was deleted successfully, FALSE if not.

resource_view_list	<i>List resource views for a resource</i>
--------------------	---

Description

List resource views for a resource

Usage

```
resource_view_list(
  id,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

id	(character or ckan_resource) Resource identifier.
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
pkg <- package_show("sample-dataset")
resource_view_list(pkg$resources[[1]]$id)

## End(Not run)
```

resource_view_reorder	<i>Reorder resource views</i>
-----------------------	-------------------------------

Description

Reorder resource views

Usage

```
resource_view_reorder(
  id,
  order,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

id	(character or ckan_resource) Resource identifier.
order	(character vector) View IDs in the desired order.
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

resource_view_show	<i>Show a resource view</i>
--------------------	-----------------------------

Description

Show a resource view

Usage

```
resource_view_show(
  id,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

id	(character or ckan_resource_view) View identifier.
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .

key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to <code>data.frame</code> 's when possible, so the result can vary from a vector, list or <code>data.frame</code> . (required)
...	Curl args passed on to verb-POST (optional)

resource_view_update *Update an existing resource view*

Description

Update an existing resource view

Usage

```
resource_view_update(
  id,
  resource = NULL,
  title = NULL,
  description = NULL,
  config = NULL,
  filter_fields = NULL,
  filter_values = NULL,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

id	(character or ckan_resource_view) View identifier.
resource	(character or ckan_resource) Parent resource identifier.
title	(character) Title assigned to the view.
description	(character) Optional description.
config	(list) Arbitrary configuration list passed to the plugin.
filter_fields, filter_values	Optional filter parameters for filterable views.
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup

as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to <code>data.frame</code> 's when possible, so the result can vary from a vector, list or <code>data.frame</code> . (required)
...	Curl args passed on to verb-POST (optional)

resource_views	<i>Resource view management</i>
----------------	---------------------------------

Description

Helpers that wrap CKAN's `resource_view_*` endpoints.

revision_list	<i>Return a list of the IDs of the site's revisions.</i>
---------------	--

Description

Return a list of the IDs of the site's revisions.

Usage

```
revision_list(
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to <code>data.frame</code> 's when possible, so the result can vary from a vector, list or <code>data.frame</code> . (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
revision_list()
revision_list(as = "table")
revision_list(as = "json")

## End(Not run)
```

servers*CKAN server URLs and other info*

Description

CKAN server URLs and other info

Usage

```
servers()
```

Details

CKAN instances from <https://github.com/ckan/ckan-instances/>

Examples

```
## Not run:
servers()
ckan_info(servers()[5])

# what version is each CKAN server running
out <- lapply(servers()[1:6], function(w) {
  ckan_version(w)$version_num
})

## End(Not run)
```

src_ckan*Connect to CKAN with dplyr*

Description

Use `src_ckan` to connect to an existing CKAN instance and `tbl` to connect to tables within that CKAN based on the DataStore Data API.

Usage

```
src_ckan(url)
```

Arguments

`url` the url of the CKAN instance

Examples

```
## Not run:
library("dplyr")

# To connect to a CKAN instance first create a src:
my_ckan <- src_ckan("http://demo.ckan.org")

# List all tables in the CKAN instance
db_list_tables(my_ckan$con)

# Then reference a tbl within that src
my_tbl <- tbl(src = my_ckan, name = "44d7de5f-7029-4f3a-a812-d7a70895da7d")

# You can use the dplyr verbs with my_tbl. For example:
dplyr::filter(my_tbl, GABARITO == "C")

## End(Not run)
```

tag_autocomplete	<i>Autocomplete tag names.</i>
------------------	--------------------------------

Description

Autocomplete tag names.

Usage

```
tag_autocomplete(
  q,
  vocabulary_id = NULL,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

q	(character) Partial tag name to search for. Required.
vocabulary_id	(character) Restrict matches to a specific vocabulary.
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
tag_autocomplete(q = "ckan")

## End(Not run)
```

tag_create	Create a tag
------------	--------------

Description

IMPORTANT: You must be a sysadmin to create vocabulary tags.

Usage

```
tag_create(
  name,
  vocabulary_id,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

name	(character) The name for the new tag, a string between 2 and 100 characters long containing only alphanumeric characters and -, _ and ., e.g. 'Jazz'
vocabulary_id	(character) The id of the vocabulary that the new tag should be added to, e.g. the id of vocabulary 'Genre'
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
ckanr_setup(
  url = "https://demo.ckan.org/",
  key = Sys.getenv("CKAN_DEMO_KEY")
)
```

```

tag_create(name = "TestTag1", vocabulary_id = "Testing1")

## End(Not run)

```

tag_list	<i>List tags.</i>
----------	-------------------

Description

List tags.

Usage

```

tag_list(
  query = NULL,
  vocabulary_id = NULL,
  all_fields = FALSE,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)

```

Arguments

query	(character) A tag name query to search for, if given only tags whose names contain this string will be returned
vocabulary_id	(character) The id or name of a vocabulary, if given, only tags that belong to this vocabulary will be returned
all_fields	(logical) Return full tag dictionaries instead of just names. Default: FALSE
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

Examples

```

## Not run:
# list all tags
tag_list()

# search for a specific tag

```

```

tag_list(query = "aviation")

# all fields
tag_list(all_fields = TRUE)

# give back different data formats
tag_list("aviation", as = "json")
tag_list("aviation", as = "table")

## End(Not run)

```

tag_search	<i>Search tags.</i>
------------	---------------------

Description

Search tags.

Usage

```

tag_search(
  query = NULL,
  vocabulary_id = NULL,
  offset = 0,
  limit = 31,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)

```

Arguments

query	(character) A tag name query to search for, if given only tags whose names contain this string will be returned; one or more search strings
vocabulary_id	(character) The id or name of a vocabulary, if give only tags that belong to this vocabulary will be returned
offset	(numeric) Where to start getting activity items from (optional, default: 0)
limit	(numeric) The maximum number of activities to return (optional, default: 31)
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
tag_search(query = "ta")
tag_search(query = c("ta", "al"))

# different formats back
tag_search(query = "ta", as = "json")
tag_search(query = "ta", as = "table")

## End(Not run)
```

tag_show	<i>Show a tag.</i>
----------	--------------------

Description

Show a tag.

Usage

```
tag_show(
  id,
  include_datasets = FALSE,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

id	(character) The name or id of the tag
include_datasets	include a list of up to 1000 of the tag's datasets. Limit 1000 datasets, use package_search() for more. (optional, default: FALSE)
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
# get tags with tag_list()
tags <- tag_list()
tags[[30]]$id

# show a tag
(x <- tag_show(tags[[30]]$id))

# give back different data formats
tag_show(tags[[30]]$id, as = "json")
tag_show(tags[[30]]$id, as = "table")

## End(Not run)
```

task_status

CKAN task status maintenance

Description

These helpers wrap the sysadmin-only task status endpoints that CKAN uses to track background operations (eg DataPusher runs).

Usage

```
task_status_show(
  id = NULL,
  entity_id = NULL,
  entity_type = NULL,
  task_type = NULL,
  task_key = NULL,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)

task_status_update(
  id = NULL,
  entity_id = NULL,
  entity_type = NULL,
  task_type = NULL,
  task_key = NULL,
  value = NULL,
  state = NULL,
  last_updated = NULL,
  error = NULL,
  url = get_default_url(),
```

```

    key = get_default_key(),
    as = "list",
    ...
)

task_status_update_many(
    data,
    url = get_default_url(),
    key = get_default_key(),
    as = "list",
    ...
)

task_status_delete(
    id = NULL,
    entity_id = NULL,
    entity_type = NULL,
    task_type = NULL,
    task_key = NULL,
    url = get_default_url(),
    key = get_default_key(),
    as = "list",
    ...
)

```

Arguments

id	(character) Task status identifier to inspect or mutate.
entity_id	(character) ID of the dataset/resource related to the task.
entity_type	(character) CKAN domain object type (e.g., "dataset").
task_type	(character) Task namespace such as "datapusher".
task_key	(character) Additional task key (e.g., queue name).
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to <code>data.frame</code> 's when possible, so the result can vary from a vector, list or <code>data.frame</code> . (required)
...	Curl args passed on to verb-POST (optional)
value	(character) Free-form value, often notes or payload references.
state	(character) Task state such as "queued", "running", or "finished".
last_updated	(character) Timestamp string for the last state change.
error	(character) Optional serialized error blob attached to the task.
data	(list) For <code>task_status_update_many()</code> , a list of task status dictionaries to update.

Examples

```
## Not run:
ckanr_setup(url = "https://demo.ckan.org/", key = "my-ckan-key")
task_status_show(entity_type = "dataset", entity_id = "my-dataset")
task_status_update(
  entity_id = "my-dataset",
  entity_type = "dataset",
  task_type = "datapusher",
  task_key = "default",
  value = "queued",
  state = "running"
)

## End(Not run)
```

term_translation	<i>Term translation helpers</i>
------------------	---------------------------------

Description

Manage localized UI strings stored in CKAN's term translation table.

Usage

```
term_translation_show(
  terms,
  lang_codes = NULL,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)

term_translation_update(
  term,
  term_translation,
  lang_code,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)

term_translation_update_many(
  data,
  url = get_default_url(),
  key = get_default_key(),
```

```

    as = "list",
    ...
)

```

Arguments

terms	(character) Vector of source terms.
lang_codes	(character) Language codes to include.
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)
term	(character) Source term.
term_translation	(character) Translation value.
lang_code	(character) Target language code.
data	(list) List of translation dictionaries with term, term_translation, and lang_code entries.

Examples

```

## Not run:
ckanr_setup(url = "https://demo.ckan.org/", key = "my-ckan-key")
term_translation_show(c("License", "Dataset"), lang_codes = "fr")
term_translation_update(term = "License", term_translation = "Licence", lang_code = "fr")

## End(Not run)

```

user_activity_list	<i>Return a list of a user's activities</i>
--------------------	---

Description

Return a list of a user's activities

Usage

```

user_activity_list(
  id,
  offset = 0,
  limit = 31,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)

```

Arguments

<code>id</code>	(character) User identifier.
<code>offset</code>	(numeric) Where to start getting activity items from (optional, default: 0)
<code>limit</code>	(numeric) The maximum number of activities to return (optional, default: 31)
<code>url</code>	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
<code>key</code>	A privileged CKAN API key, Default: your key set with ckanr_setup
<code>as</code>	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to <code>data.frame</code> 's when possible, so the result can vary from a vector, list or <code>data.frame</code> . (required)
<code>...</code>	Curl args passed on to verb-POST (optional)

Examples

```

## Not run:
# Setup
ckanr_setup(
  url = "https://demo.ckan.org/",
  key = getOption("ckan_demo_key")
)

# list package activity
user_activity_list("sckotttie")

# input a ckan_user object
(x <- user_show("sckotttie"))
user_activity_list(x)

# output different data formats
user_activity_list(x, as = "table")
user_activity_list(x, as = "json")

## End(Not run)

```

user_create

Create a user.

Description

Create a user.

Usage

```
user_create(
  name,
  email,
  password,
  id = NULL,
  fullname = NULL,
  about = NULL,
  openid = NULL,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

name	(character) the name of the new user, a string between 2 and 100 characters in length, containing only lowercase alphanumeric characters, - and _ (required)
email	(character) the email address for the new user (required)
password	(character) the password of the new user, a string of at least 4 characters (required)
id	(character) the id of the new user (optional)
fullname	(character) user full name
about	(character) a description of the new user (optional)
openid	(character) an openid (optional)
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)

References

http://docs.ckan.org/en/latest/api/index.html#ckan.logic.action.create.user_create

Examples

```
## Not run:
# Setup
ckanr_setup(
  url = "https://data-demo.dpaw.wa.gov.au",
  key = "824e7c50-9577-4bfa-bf32-246ebed1a8a2"
)

# create a user
user_create(
  name = "stacy", email = "stacyaaaaa.com",
  password = "helloworld"
)

## End(Not run)
```

user_delete	<i>Delete a user.</i>
-------------	-----------------------

Description

Delete a user.

Usage

```
user_delete(
  id,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

- id (character) the id of the new user (required)
- url Base url to use. Default: <https://demo.ckan.org/> See also [ckanr_setup](#) and [get_default_url](#).
- key A privileged CKAN API key, Default: your key set with [ckanr_setup](#)
- as (character) One of list (default), table, or json. Parsing with table option uses `jsonlite::fromJSON(..., simplifyDataFrame = TRUE)`, which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
- ... Curl args passed on to [verb-POST](#) (optional)

Value

(logical) TRUE when the user is successfully deleted

References

http://docs.ckan.org/en/latest/api/index.html#ckan.logic.action.delete.user_delete

Examples

```
## Not run:
# Setup
ckanr_setup(
  url = "https://data-demo.dpaw.wa.gov.au",
  key = "824e7c50-9577-4bfa-bf32-246ebed1a8a2"
)

# create a user
res <- user_delete(
  name = "stacy", email = "stacyaaaaa.com",
  password = "helloworld"
)

# then, delete a user
user_delete(id = "stacy")

## End(Not run)
```

user_invite	<i>Invite a user to a group</i>
-------------	---------------------------------

Description

Invite a user to a group

Usage

```
user_invite(
  email,
  group_id,
  role = "member",
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

email	(character) Email address for the invitee.
group_id	(character or ckan_group) Group identifier.
role	(character) Desired role ("member", "editor", "admin").
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to <code>data.frame</code> 's when possible, so the result can vary from a vector, list or <code>data.frame</code> . (required)
...	Curl args passed on to verb-POST (optional)

user_list	<i>Return a list of the site's user accounts.</i>
-----------	---

Description

Return a list of the site's user accounts.

Usage

```
user_list(
  q = NULL,
  order_by = NULL,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

q	(character) Restrict the users returned to those whose names contain a string
order_by	(character) Which field to sort the list by (optional, default: 'name')
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to <code>data.frame</code> 's when possible, so the result can vary from a vector, list or <code>data.frame</code> . (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
# all users
user_list()

# search for a user
user_list(q = "j")

# different data formats
user_list(q = "j", as = "table")
user_list(q = "j", as = "json")

## End(Not run)
```

user_show	<i>Show a user.</i>
-----------	---------------------

Description

Show a user.

Usage

```
user_show(
  id,
  user_obj = NULL,
  include_datasets = FALSE,
  include_num_followers = FALSE,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

Arguments

id	(character) Package identifier.
user_obj	(user dictionary) The user dictionary of the user (optional)
include_datasets	(logical) Include a list of datasets the user has created. If it is the same user or a sysadmin requesting, it includes datasets that are draft or private. (optional, default:False, limit:50)
include_num_followers	(logical) Include the number of followers the user has (optional, default:False)
url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .

key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to <code>data.frame</code> 's when possible, so the result can vary from a vector, list or <code>data.frame</code> . (required)
...	Curl args passed on to verb-POST (optional)

Examples

```
## Not run:
# Setup
ckanr_setup(url = "https://demo.ckan.org/", key = getOption("ckan_demo_key"))

# show user
user_show("sckotttie")

# include datasets
user_show("sckotttie", include_datasets = TRUE)

# include datasets
user_show("sckotttie", include_num_followers = TRUE)

## End(Not run)
```

vocabulary

Manage CKAN vocabularies.

Description

CKAN restricts vocabulary creation, updates, and deletions to sysadmin users. Each helper wraps the matching `/api/3/action/vocabulary_*` endpoint.

Usage

```
vocabulary_list(
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)

vocabulary_show(
  id,
  include_datasets = FALSE,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)
```

```

)

vocabulary_create(
  name,
  tags = NULL,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)

vocabulary_update(
  id,
  name = NULL,
  tags = NULL,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)

vocabulary_delete(
  id,
  url = get_default_url(),
  key = get_default_key(),
  as = "list",
  ...
)

```

Arguments

url	Base url to use. Default: https://demo.ckan.org/ See also ckanr_setup and get_default_url .
key	A privileged CKAN API key, Default: your key set with ckanr_setup
as	(character) One of list (default), table, or json. Parsing with table option uses <code>jsonlite::fromJSON(..., simplifyDataFrame = TRUE)</code> , which attempts to parse data to data.frame's when possible, so the result can vary from a vector, list or data.frame. (required)
...	Curl args passed on to verb-POST (optional)
id	(character or ckan_vocabulary) Vocabulary id or name. Can also be an existing ckan_vocabulary object.
include_datasets	(logical) Return datasets owned by the vocabulary. Only applies to <code>vocabulary_show()</code> .
name	(character) Unique vocabulary name.
tags	(list) Optional list of tag objects, each containing at least a name field. See https://docs.ckan.org/en/latest/api/ for the full structure.

Examples

```
## Not run:
ckanr_setup(url = "https://demo.ckan.org/", key = getOption("ckan_demo_key"))

vocab_name <- sprintf("demo_vocab_%s", sample.int(1e6, 1))
vocab <- vocabulary_create(name = vocab_name)

vocabulary_show(vocab$id)
vocabulary_update(vocab, name = paste0(vocab_name, "_updated"))
vocabulary_list()
vocabulary_delete(vocab)

## End(Not run)
```

Index

- * **ckanr settings**
 - ckanr_settings, [24](#)
- * **package**
 - ckanr-package, [4](#)
- activity_create (activity_helpers), [6](#)
- activity_data_show (activity_helpers), [6](#)
- activity_diff (activity_helpers), [6](#)
- activity_helpers, [6](#)
- activity_show (activity_helpers), [6](#)
- am_following_user (follow_users), [39](#)
- api_token_create (api_tokens), [8](#)
- api_token_list (api_tokens), [8](#)
- api_token_revoke (api_tokens), [8](#)
- api_tokens, [8](#)
- as.ckan_group, [10](#)
- as.ckan_organization, [11](#)
- as.ckan_package, [11](#)
- as.ckan_related, [12](#)
- as.ckan_resource, [13](#)
- as.ckan_resource_view, [14](#)
- as.ckan_tag, [15](#)
- as.ckan_user, [16](#)
- as.ckan_vocabulary, [16](#)
- changes, [17](#)
- ckan_action, [18](#)
- ckan_classes, [19](#)
- ckan_fetch, [20](#)
- ckan_info, [23](#)
- ckan_version (ckan_info), [23](#)
- ckan_version(), [90](#)
- ckanr (ckanr-package), [4](#)
- ckanr-deprecated, [24](#)
- ckanr-package, [4](#)
- ckanr_settings, [24](#)
- ckanr_setup, [8](#), [9](#), [18](#), [19](#), [25](#), [28–31](#), [33](#), [34](#),
[36](#), [37](#), [39–42](#), [44](#), [45](#), [47](#), [49–54](#),
[56–60](#), [62–69](#), [71–73](#), [75–88](#), [90](#), [91](#),
[93–98](#), [100](#), [102](#), [103](#), [105–107](#),
[110–115](#), [117–121](#), [123](#), [125–128](#),
[130–133](#)
- ckanr_setup(), [5](#), [19](#), [24–26](#), [47](#), [77](#), [96](#), [102](#)
- config_option_list (config_options), [27](#)
- config_option_show (config_options), [27](#)
- config_option_update (config_options),
[27](#)
- config_options, [27](#)
- crul::HttpClient, [19](#)
- crul::proxy(), [26](#)
- crul::verb-GET, [24](#)
- crul::verb-POST, [47](#), [77](#), [96](#), [102](#)
- dashboard_activity_list, [28](#)
- dashboard_count, [29](#)
- dashboard_mark_activities_old
(activity_helpers), [6](#)
- dashboard_new_activities_count
(activity_helpers), [6](#)
- dataset_am_following (follow_objects),
[38](#)
- dataset_follow (follow_objects), [38](#)
- dataset_follower_count
(follower_counts), [40](#)
- dataset_follower_list (follower_lists),
[41](#)
- dataset_follower_count
(follower_counts), [43](#)
- dataset_follower_list (follower_lists),
[44](#)
- dataset_purge, [30](#)
- dataset_unfollow (follow_objects), [38](#)
- diagnostics, [31](#)
- dplyr-interface (src_ckan), [116](#)
- ds_create, [32](#)
- ds_create(), [5](#)
- ds_create_dataset, [33](#)
- ds_create_dataset(), [5](#), [24](#)
- ds_search, [35](#)
- ds_search(), [5](#)

- ds_search_sql, 37
- ds_search_sql(), 5
- follow_objects, 38
- follow_user (follow_users), 39
- follow_users, 39
- followee_count (followee_counts), 40
- followee_counts, 40
- followee_list (followee_lists), 41
- followee_lists, 41
- follower_counts, 43
- follower_lists, 44
- fromJSON, 21
- get_default_key (ckanr_settings), 24
- get_default_key(), 5, 25
- get_default_url, 8, 9, 18, 28–31, 33, 34, 36, 37, 39–42, 44, 45, 47, 49–54, 56–60, 62–69, 71–73, 75, 76, 78–88, 90, 91, 93–95, 97, 98, 100, 102, 103, 105–107, 110–115, 117–121, 123, 125–128, 130, 131, 133
- get_default_url (ckanr_settings), 24
- get_default_url(), 19, 24, 25, 47, 77, 96, 102
- get_test_behaviour (ckanr_settings), 24
- get_test_did (ckanr_settings), 24
- get_test_did(), 25
- get_test_gid (ckanr_settings), 24
- get_test_gid(), 25
- get_test_key (ckanr_settings), 24
- get_test_key(), 25
- get_test_oid (ckanr_settings), 24
- get_test_rid (ckanr_settings), 24
- get_test_rid(), 25
- get_test_url (ckanr_settings), 24
- get_test_url(), 25
- group_activity_list (activity_helpers), 6
- group_am_following (follow_objects), 38
- group_create, 45
- group_delete, 47
- group_follow (follow_objects), 38
- group_followee_count (followee_counts), 40
- group_followee_list (followee_lists), 41
- group_follower_count (follower_counts), 43
- group_follower_list (follower_lists), 44
- group_list, 48
- group_list(), 5, 46, 75
- group_list_authz, 49
- group_member_create, 50
- group_member_delete, 51
- group_patch, 52
- group_show, 53
- group_show(), 10
- group_unfollow (follow_objects), 38
- group_update, 54
- help_show (diagnostics), 31
- is.ckan_group (as.ckan_group), 10
- is.ckan_organization (as.ckan_organization), 11
- is.ckan_package (as.ckan_package), 11
- is.ckan_related (as.ckan_related), 12
- is.ckan_resource (as.ckan_resource), 13
- is.ckan_resource_view (as.ckan_resource_view), 14
- is.ckan_tag (as.ckan_tag), 15
- is.ckan_user (as.ckan_user), 16
- is.ckan_vocabulary (as.ckan_vocabulary), 16
- job_cancel (job_queue), 55
- job_clear (job_queue), 55
- job_list (job_queue), 55
- job_queue, 55
- job_show (job_queue), 55
- license_list, 56
- member_create, 57
- member_delete, 58
- member_list, 59
- member_roles_list, 60
- membership, 60
- organization_activity_list (activity_helpers), 6
- organization_create, 61
- organization_create(), 5
- organization_delete, 62
- organization_list, 63
- organization_list(), 5, 75
- organization_list_for_user, 64
- organization_member_create, 65
- organization_member_delete, 66

organization_purge, 67
organization_show, 68
organization_show(), 5, 11

package_activity_list, 69
package_collaborator_create, 70
package_collaborator_delete, 71
package_collaborator_list, 72
package_collaborator_list_for_user, 73
package_collaborators, 73
package_create, 74
package_create(), 5
package_create_default_resource_views,
 76
package_dataset_extras, 77
package_delete, 77
package_list, 78
package_list_current, 79
package_owner_org_update, 80
package_patch, 80
package_relationship_create, 82
package_relationship_create(), 85
package_relationship_delete, 83
package_relationship_update, 83
package_relationships, 84
package_relationships_list, 85
package_resource_reorder, 86
package_revise, 87
package_revision_list, 88
package_search, 89
package_search(), 121
package_show, 91
package_show(), 12
package_update, 92
ping, 93

read.csv, 21
read.csv(), 5
read.table, 21
read_excel, 21
read_html, 21
read_parquet, 21
read_xml, 21
recently_changed_packages_activity_list
 (activity_helpers), 6
related_create, 94
related_delete, 96
related_list, 97
related_show, 98

related_show(), 13
resource_create, 99
resource_create(), 5, 24, 32, 75
resource_create_default_resource_views,
 101
resource_delete, 102
resource_patch, 103
resource_search, 104
resource_show, 105
resource_show(), 13
resource_update, 107
resource_view_clear, 110
resource_view_create, 110
resource_view_delete, 111
resource_view_list, 112
resource_view_reorder, 112
resource_view_show, 113
resource_view_show(), 14
resource_view_update, 114
resource_views, 115
revision_list, 115

send_email_notifications
 (activity_helpers), 6
servers, 116
servers(), 5
src_ckan, 116
st_read, 21
status_show (diagnostics), 31

tag_autocomplete, 117
tag_create, 118
tag_create(), 75
tag_list, 119
tag_list(), 5
tag_search, 120
tag_show, 121
tag_show(), 15
task_status, 122
task_status_delete (task_status), 122
task_status_show (task_status), 122
task_status_update (task_status), 122
task_status_update_many (task_status),
 122
term_translation, 124
term_translation_show
 (term_translation), 124
term_translation_update
 (term_translation), 124

`term_translation_update_many`
 (`term_translation`), [124](#)

`unfollow_user` (`follow_users`), [39](#)

`user_activity_list`, [125](#)

`user_create`, [127](#)

`user_delete`, [128](#)

`user_follower_count` (`follower_counts`),
 [40](#)

`user_follower_list` (`follower_lists`), [44](#)

`user_follower_count` (`follower_counts`),
 [43](#)

`user_follower_list` (`follower_lists`), [44](#)

`user_invite`, [129](#)

`user_list`, [130](#)

`user_list()`, [5](#)

`user_show`, [131](#)

`user_show()`, [16](#)

`vocabulary`, [132](#)

`vocabulary_create` (`vocabulary`), [132](#)

`vocabulary_delete` (`vocabulary`), [132](#)

`vocabulary_list` (`vocabulary`), [132](#)

`vocabulary_show` (`vocabulary`), [132](#)

`vocabulary_update` (`vocabulary`), [132](#)