Package ‘ckanr’

August 29, 2016

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

Description  Client for ‘CKAN’ ‘API’ (http://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  0.1.0
License  MIT + file LICENSE
LazyData  true

URL  https://github.com/ropensci/ckanr

BugReports  http://www.github.com/ropensci/ckanr/issues

VignetteBuilder  knitr

Imports  methods, stats, utils, httr (>= 1.0.0), jsonlite (>= 0.9.17), magrittr

Suggests  knitr, maptools, readxl, testthat, xml2

NeedsCompilation  no

Author  Scott Chamberlain [aut, cre],
        Imanuel Costigan [ctb],
        Wush Wu [ctb],
        Florian Mayer [ctb]

Maintainer  Scott Chamberlain <myrmecocystus@gmail.com>

Repository  CRAN

Date/Publication  2015-10-22 08:24:21

R topics documented:
ckanr-package .......................................................... 3
as.ckan_group ......................................................... 4
as.ckan_organization ............................................... 5
as.ckan_package ..................................................... 6
as.ckan_related ..................................................... 6
<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>as.ckan_resource</td>
<td>7</td>
</tr>
<tr>
<td>as.ckan_tag</td>
<td>8</td>
</tr>
<tr>
<td>as.ckan_user</td>
<td>9</td>
</tr>
<tr>
<td>changes</td>
<td>9</td>
</tr>
<tr>
<td>ckanr_settings</td>
<td>10</td>
</tr>
<tr>
<td>ckanr_setup</td>
<td>11</td>
</tr>
<tr>
<td>ckan_classes</td>
<td>12</td>
</tr>
<tr>
<td>ckan_info</td>
<td>13</td>
</tr>
<tr>
<td>dashboard_activity_list</td>
<td>14</td>
</tr>
<tr>
<td>dashboard_count</td>
<td>15</td>
</tr>
<tr>
<td>ds_create</td>
<td>16</td>
</tr>
<tr>
<td>ds_create_dataset</td>
<td>17</td>
</tr>
<tr>
<td>ds_search</td>
<td>18</td>
</tr>
<tr>
<td>ds_search_sql</td>
<td>19</td>
</tr>
<tr>
<td>fetch</td>
<td>20</td>
</tr>
<tr>
<td>group_create</td>
<td>21</td>
</tr>
<tr>
<td>group_delete</td>
<td>23</td>
</tr>
<tr>
<td>group_list</td>
<td>23</td>
</tr>
<tr>
<td>group_patch</td>
<td>24</td>
</tr>
<tr>
<td>group_show</td>
<td>25</td>
</tr>
<tr>
<td>group_update</td>
<td>26</td>
</tr>
<tr>
<td>license_list</td>
<td>27</td>
</tr>
<tr>
<td>organization_create</td>
<td>28</td>
</tr>
<tr>
<td>organization_list</td>
<td>29</td>
</tr>
<tr>
<td>organization_show</td>
<td>30</td>
</tr>
<tr>
<td>package_activity_list</td>
<td>31</td>
</tr>
<tr>
<td>package_create</td>
<td>32</td>
</tr>
<tr>
<td>package_delete</td>
<td>34</td>
</tr>
<tr>
<td>package_list</td>
<td>35</td>
</tr>
<tr>
<td>package_list_current</td>
<td>36</td>
</tr>
<tr>
<td>package_patch</td>
<td>37</td>
</tr>
<tr>
<td>package_revision_list</td>
<td>38</td>
</tr>
<tr>
<td>package_search</td>
<td>39</td>
</tr>
<tr>
<td>package_show</td>
<td>40</td>
</tr>
<tr>
<td>package_update</td>
<td>41</td>
</tr>
<tr>
<td>ping</td>
<td>42</td>
</tr>
<tr>
<td>related_create</td>
<td>43</td>
</tr>
<tr>
<td>related_delete</td>
<td>44</td>
</tr>
<tr>
<td>related_list</td>
<td>45</td>
</tr>
<tr>
<td>related_show</td>
<td>45</td>
</tr>
<tr>
<td>related_update</td>
<td>46</td>
</tr>
<tr>
<td>resource_create</td>
<td>47</td>
</tr>
<tr>
<td>resource_delete</td>
<td>49</td>
</tr>
<tr>
<td>resource_patch</td>
<td>50</td>
</tr>
<tr>
<td>resource_search</td>
<td>51</td>
</tr>
<tr>
<td>resource_show</td>
<td>52</td>
</tr>
<tr>
<td>resource_update</td>
<td>53</td>
</tr>
<tr>
<td>revision_list</td>
<td>55</td>
</tr>
</tbody>
</table>
ckanr-package

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

Document for the CKAN API is at http://docs.ckan.org/en/latest/api/index.html. We’ll always be following the lastest 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 (http://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>

---

**Description**

`ckan_group class helpers`

**Usage**

```python
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

```r
## Not run:
ckanr_setup(url = "http://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

```r
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

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

(orgs <- organization_list())
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

Examples

## Not run:
ckanr_setup(url = "http://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("0699f475-6978-473a-8448-42585074b6f1")

# gives back itself
(x <- as.ckan_package("0699f475-6978-473a-8448-42585074b6f1"))

as.ckan_package(x)

## End(Not run)

as.ckan_related  ckan_related class helpers

Description

ckan_related class helpers

Usage

as.ckan_related(x, ...)

is.ckan_related(x)
as.ckan_resource

**Arguments**

- `x`: Variety of things, character, list, or ckan_related class object
- `...`: Further args passed on to `related_show` if character given

**Examples**

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

(x <- package_create("foobbbbarrrr")) %>%
  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)
```

---

**Description**

ckan_resource class helpers

**Usage**

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

```r
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**

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

(resrcs <- resource_search(q = 'name:data'))
resrcs$results
resrcs$results[[3]]
```
as.ckan_tag
c

c.ckan_tag

c.ckan_resource(resrc$sresults[[3]]$id)

c.ckan_resource(x)

## End(Not run)

c.ckan_tag class helpers

**Description**

c.ckan_tag class helpers

**Usage**

as.ckan_tag(x, ...)

is.ckan_tag(x)

**Arguments**

x

Variety of things, character, list, or c.ckan_tag class object

... Further args passed on to tag_show if character given

**Examples**

## Not run:

ckanr_setup(url = "http://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

**Description**

ckan_user class helpers

**Usage**

```r
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**

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

(usr <- user_list())
usr$results
usr$results[[3]]

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

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

## 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**

```r
changes(offset = 0, limit = 31, url = get_default_url(), 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: http://data.techno-science.ca. See also ckanr_setup and get_default_url.
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)

Examples

## Not run:
changes()
changes(as = 'json')
changes(as = 'table')

## End(Not run)

ckanr_settings Get or set ckanr CKAN settings

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. testBehaviour 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 Configure default CKAN settings

Description

Configure default CKAN settings

Usage

ckanr_setup(url = "http://data.techno-science.ca/", key = NULL, test_url = NULL, test_key = NULL, test_did = NULL, test_rid = NULL, test_gid = NULL, test_oid = NULL, test_behaviour = NULL)

Arguments

- **url**: A CKAN URL (optional), default: "http://data.techno-science.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.
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

```python
# CKAN users without admin/editor privileges could run:
ckanr_setup(url = "http://data.techno-science.ca/"

# Privileged CKAN editor/admin users can run:
ckanr_setup(url = "http://data.techno-science.ca/", key = "some-CKAN-API-key")

# ckanR developers/testers can run:
ckanr_setup(url = "http://data.techno-science.ca/", 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",
  testBehaviour = "FAIL")

# Not specifying the default CKAN URL will reset the CKAN URL to its default
# "http://data.techno-science.ca/":
ckanr_setup()
```

---

**ckan_classes**

**ckan S3 classes**

Description

ckan S3 classes

The classes

- `ckan_package` - CKAN package
- `ckan_resource` - CKAN resource
- `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_info

Get information on a CKAN server

Description

Get information on a CKAN server

Usage

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

Arguments

url Base url to use. Default: http://data.techno-science.ca. See also ckanr_setup and get_default_url. (required)
... Curl args passed on to GET (optional)

Examples

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

## 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, key = get_default_key(),
url = get_default_url(), 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
- `key` A privileged CKAN API key. Default: your key set with `ckanr_setup`
- `url` Base url to use. Default: `http://data.techno-science.ca`. See also `ckanr_setup` and `get_default_url`.
- `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 `POST` (optional)

**Examples**

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

# get activity
(res <- dashboard_activity_list())
```

## End(Not run)
dashboard_count

Number of new activities of an authorized user

Description

Number of new activities of an authorized user

Usage

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

Arguments

- **key**: A privileged CKAN API key. Default: your key set with `ckanr_setup`
- **url**: Base url to use. Default: `http://data.techno-science.ca`. See also `ckanr_setup` and `get_default_url`.
- **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 `POST` (optional)

Details

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

Examples

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

# count
dashboard_count()

## End(Not run)
```
ds_create

Add a new table to a datastore

Description

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

Usage

```r
ds_create(resource_id = NULL, resource = NULL, force = FALSE,
aliases = NULL, fields = NULL, records = NULL, primary_key = NULL,
indexes = NULL, key = get_default_key(), url = get_default_url(),
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)
- **key** A privileged CKAN API key. Default: your key set with ckanr_setup
- **url** Base url to use. Default: http://data.techno-science.ca. See also ckanr_setup and get_default_url.
- **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 POST (optional)

References


Examples

```r
## Not run:
ds_create(resource_id = "f4129802-22aa-4437-b9f9-8a8f3b7b2a53",
records = iris, force = TRUE, key = "my-api-key")
```

## 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**

```r
ds_create_dataset(package_id, name, path, key = get_default_key(),
                   url = get_default_url(), 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)
- `key` A privileged CKAN API key. Default: your key set with `ckanr_setup`
- `url` Base url to use. Default: `http://data.techno-science.ca`. See also `ckanr_setup` and `get_default_url`.
- `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 POST (optional)

**References**

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

**Examples**

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

# 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")
dataset(package_id=get_test_pid(), name="mydata",
        path=system.file("examples",
                         "actinidiaceae.csv",
                         package = "ckanr"),
```
ds_search

Datastore - search or get a dataset from CKRAN datastore

Description

Datastore - search or get a dataset from CKRAN 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(), 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: http://data.techno-science.ca. See also ckanr_setup and get_default_url.

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 POST (optional)
ds_search_sql

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
```r
## Not run:
url <- 'http://demo.ckan.org/

ds_search(resource_id = 'f4129802-22aa-4437-b9f9-8a8f3b7b2a53', url = url)
ds_search(resource_id = 'f4129802-22aa-4437-b9f9-8a8f3b7b2a53', url = url, as = "table")
ds_search(resource_id = 'f4129802-22aa-4437-b9f9-8a8f3b7b2a53', url = url, as = "json")

ds_search(resource_id = 'f4129802-22aa-4437-b9f9-8a8f3b7b2a53', url = url, limit = 1, as = "table")
ds_search(resource_id = 'f4129802-22aa-4437-b9f9-8a8f3b7b2a53', url = url, q = "a*")

## End(Not run)
```

---

**ds_search_sql**

Datastore - search or get a dataset from CKRAN datastore

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

**Usage**
ds_search_sql(sql, url = get_default_url(), as = "list", ...)

**Arguments**
- `sql` (character) A single SQL select statement. (required)
- `url` Base url to use. Default: http://data.techno-science.ca. See also ckanr_setup and get_default_url.
- `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 POST (optional)
Examples

```r
## Not run:
url <- 'http://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)
```

fetch

**Download a file**

Description

Download a file

Usage

```r
fetch(x, store = "session", path = "file", ...)
```

Arguments

- **x** URL for the file
- **store** One of session (default) or disk. session stores in R session, and disk saves the file to disk.
- **path** if store=disk, you must give a path to store file to
- **...** Curl arguments passed on to GET

Examples

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

# Excel file - requires readxl package
ckanr_setup()
res <- resource_show(id = "f4f871ae-139f-4acd-a700-8a08a1a04f95", as = "table")
head(fetch(res$url))

# XML file
ckanr_setup("http://publicdata.eu/")
res <- resource_show(id = "ba447d6b-271e-42c2-965e-945f8e26b2ff", as = "table")
fetch(res$url)

# HTML file
```
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, key = get_default_key(), url = get_default_url(), 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'
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**

(list of dictionaries) The datasets (packages) that belong to the group, 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)

**groups**

(list of dictionaries) The groups to which the dataset belongs (optional), each group dictionary should have one or more of the following keys 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)

**key**

A privileged CKAN API key, Default: your key set with ckanr_setup

**url**

Base url to use. Default: http://data.techno-science.ca. See also ckanr_setup and get_default_url.

**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 POST (optional)

**Examples**

```r
## Not run:
# Setup
ckanr_setup(url = "http://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

Delete a group

Description
Delete a group

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

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

Examples
## Not run:
# Setup
ckanr_setup(url = "http://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(), 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: http://data.techno-science.ca. See also ckanr_setup
    and get_default_url.
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 POST (optional)

Examples

## Not run:
group_list()
group_list(as = 'json')
group_list(as = 'table')

## End(Not run)

---

group_patch  Update a group's metadata

Description

Update a group’s metadata

Usage

group_patch(x, id, 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 (required)
- **key** A privileged CKAN API key. Default: your key set with `ckanr_setup`
- **url** Base url to use. Default: `http://data.techno-science.ca`. See also `ckanr_setup` and `get_default_url`.
- **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)

Examples

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

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

# 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)
```

Description

Show a package

Usage

```r
group_show(id, include_datasets = TRUE, url = get_default_url(),
    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: http://data.techno-science.ca. See also ckanr_setup and get_default_url.
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 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

```r
## 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

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

Arguments

x (list) A list with key-value pairs
id (character) Package identifier
url Base url to use. Default: http://data.techno-science.ca. 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 POST (optional)

Examples

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

# First, create a group
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)

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

Description

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

Usage

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

organization_create

Arguments

id  (character) Package identifier.
url  Base url to use. Default: http://data.techno-science.ca. See also ckanr_setup and get_default_url.

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 POST (optional)

Examples

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

## End(Not run)

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,
key = get_default_key(), url = get_default_url(), 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)

**key**  
A privileged CKAN API key. Default: your key set with ckanr_setup

**url**  
Base url to use. Default: http://data.techno-science.ca. See also ckanr_setup and get_default_url.

**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 POST (optional)

---

**organization_list**  
List organization

**Description**  
List organization

**Usage**

organization_list(order_by = c("name", "package"), decreasing = FALSE, organizations = NULL, all_fields = TRUE, url = get_default_url(), as = "list", ...)
organization_show

Arguments

order_by (character, only the first element is used). The field to sort the list by, must be name or packages.
decreasing (logical). Is the sort-order is decreasing or not.
organizations (character or NULL). A list of names of the organizations to return. NULL returns all organizations.
alld_fields (logical). Return the name or all fields of the object.
url Base url to use. Default: http://data.techno-science.ca. See also ckanr_setup and get_default_url.
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)
...

Examples

```r

## Not run:
ckanr_setup(url = "http://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_show Show an organization

Description

Show an organization

Usage

organization_show(id, include_datasets = FALSE, url = get_default_url(),
key = get_default_key(), as = "list", ...)
package_activity_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: http://data.techno-science.ca. 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 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

```r
# Not run:
ckanr_setup(url = "http://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

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

Arguments

- **id** (character) Package identifier.
- **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: `http://data.techno-science.ca`. See also `ckanr_setup` and `get_default_url`.
- **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 POST (optional)

Examples

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

# create a package
(res <- package_create("owls6"))

# 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_create**  
Create a package

**Description**

Create a package
package_create

Usage

```python
package_create(name = NULL, title = NULL, 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, key = get_default_key(), url = get_default_url(), 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)
- **author** (character) the name of the dataset’s author (optional)
- **author_email** (character) the email address of the dataset’s author (optional)
- **maintainer** (character) the name of the dataset’s maintainer (optional)
- **maintainer_email** (character) the email address of the dataset’s maintainer (optional)
- **license_id** (license id string) - the id of the dataset’s license, see license_list() for available values (optional)
- **notes** (character) a description of the dataset (optional)
- **package_url** (character) a URL for the dataset’s source (optional)
- **version** (string, no longer than 100 characters) - (optional)
- **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')
- **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
- **resources** (list of resource dictionaries) - the dataset’s resources, see resource_create() for the format of resource dictionaries (optional)
- **tags** (list of tag dictionaries) - the dataset’s tags, see tag_create() for the format of tag dictionaries (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)
- **relationships_as_object** (list of relationship dictionaries) - see package_relationship_create() for the format of relationship dictionaries (optional)
package_delete

Delete a package

Description

Delete a package

Usage

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

Arguments

- **id** (character) The id of the package. Required.
- **key** A privileged CKAN API key. Default: your key set with `ckanr_setup`
- **url** Base url to use. Default: `http://data.techno-science.ca`. See also `ckanr_setup` and `get_default_url`.
  
  ... Curl args passed on to POST (optional)

Examples

```r
## Not run:
# Setup
crnr_setup(url = "http://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

List datasets.

Description

List datasets.

Usage

```r
package_list(offset = 0, limit = 31, url = get_default_url(),
   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: `http://data.techno-science.ca`. See also `ckanr_setup` and `get_default_url`.
- **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 POST (optional)
package_list_current

List current packages with resources.

Usage

package_list_current(offset = 0, limit = 31, url = get_default_url(),
    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: http://data.techno-science.ca. See also ckanr_setup
        and get_default_url.
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 POST (optional)

Examples

## Not run:
package_list_current()
package_list_current(as = 'json')
package_list_current(as = 'table')

## End(Not run)
package_patch

Update a package’s metadata

**Description**

Update a package’s metadata

**Usage**

```r
defaut_key(), url = get_default_url(),
    as = "list", ...)
```

**Arguments**

- `x` (list) A list with key-value pairs
- `id` (character) Resource ID to update (required)
- `key` A privileged CKAN API key, Default: your key set with `ckanr_setup`
- `url` Base url to use. Default: `http://data.techno-science.ca`. See also `ckanr_setup` and `get_default_url`.
- `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 `POST` (optional)

**Examples**

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

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

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

# Make some changes
x <- list(title = "hello there world! bye")
package_patch(x, id = res$id)
```

## 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(), as = "list", ...)

Arguments

id  
(character) Package identifier.

url  
Base url to use. Default: http://data.techno-science.ca. See also ckanr_setup and get_default_url.

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 POST (optional)

Examples

## Not run:
# Setup
ckanr_setup(url = "http://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

Search for packages.

Description

Search for packages.

Usage

package_search(q = "*:*", fq = NULL, sort = NULL, rows = NULL,
                start = NULL, facet = FALSE, facet.limit = NULL, facet.field = NULL,
                url = get_default_url(), 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. De-
               fault: 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.
url       Base url to use. Default: http://data.techno-science.ca. See also ckanr_setup
          and get_default_url.
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 POST (optional)
package_show

Examples

```r
## Not run:
ckanr_setup(url = "http://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)
```

Description

Show a package.

Usage

```r
package_show(id, use_default_schema = FALSE, url = get_default_url(),
              as = "list", ...)
```

Arguments

- `id` (character) Package identifier.
- `use_default_schema` (logical) Use default package schema instead of a custom schema defined with an IDatasetForm plugin. Default: FALSE
- `url` Base url to use. Default: http://data.techno-science.ca. See also ckanr_setup and get_default_url.
- `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 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.
package_update

Examples

```r
## Not run:
# Setup
ckanr_setup(url = "http://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  Update a package

Description
Update a package

Usage

```r
package_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: `http://data.techno-science.ca`. 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 POST (optional)
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(), as = "logical", ...)

Arguments
url  
Base url to use. Default: http://data.techno-science.ca. See also ckanr_setup and get_default_url.

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 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, key = get_default_key(), url = get_default_url(), 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
- key: A privileged CKAN API key. Default: your key set with ckanr_setup
- url: Base url to use. Default: http://data.techno-science.ca. See also ckanr_setup and get_default_url.
- 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)

Examples

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

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

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

Delete a related item.

Description

Delete a related item.

Usage

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

Arguments

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

Examples

## Not run:
# Setup
ckanr_setup(url = "http://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  

List related items

Description
List related items

Usage
related_list(offset = 0, limit = 31, url = get_default_url(), 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: http://data.techno-science.ca. See also ckanr_setup
  and get_default_url.
- 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 POST (optional)

Examples
```r
## Not run:
related_list()
related_list(as = 'json')
related_list(as = 'table')

## End(Not run)
```

related_show  

Show a related item

Description
Show a related item

Usage
related_show(id, url = get_default_url(), as = "list", ...)
**related_update**

Update a related item

---

**Description**

Update a related item

**Usage**

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

---

**Arguments**

- **id** (character) Related item identifier.
- **url** Base url to use. Default: `http://data.techno-science.ca`. See also `ckanr_setup` and `get_default_url`.
- **as** (character) One of list (default), table, or json. Parsing with table option uses `jsonlite::fromJSON(...)`, 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 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**

```r
## Not run:
#
# Setup
ckanr_setup(url = "http://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)
```

---

**related_update**

Update a related item
resource_create

Arguments

- **id** (character) id of related item to update. 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
- **key** A privileged CKAN API key, Default: your key set with ckanr_setup
- **url** Base url to use. Default: http://data.techno-science.ca. See also ckanr_setup and get_default_url.
- **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 POST (optional)

Examples

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

# create a package and related item
res <- package_create("hello-saturn2") %>%
  related_create(title = "my resource",
    type = "visualization")

# update the related item
related_update(res, title = "her resource", type = "idea")

## End(Not run)
```

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, key = get_default_key(),
                url = get_default_url(), 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 (FieldStorage needs multipart/form-data) - (optional)

key A privileged CKAN API key. Default: your key set with ckanr_setup

url Base url to use. Default: http://data.techno-science.ca. See also ckanr_setup
and get_default_url.

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 POST (optional)
resource_delete

## Examples

```r
## Not run:
# Setup
ckanr_setup(url = "http://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 = "585d7ea2-ded0-4fed-9b08-61f7e83a3cb2",
    description = "my resource",
    name = "bears",
    upload = file,
    rcurl = "http://google.com")
))

package_create("foobbbbbbarrrrr")
resource_create(description = "my resource",
    name = "bearsareus", upload = file, rcurl = "http://google.com")
```

## End(Not run)

---

### resource_delete

Delete a resource.

#### Description

Delete a resource.

#### Usage

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

#### Arguments

- **id** (character) Resource identifier.
- **key** A privileged CKAN API key. Default: your key set with `ckanr_setup`
- **url** Base url to use. Default: `http://data.techno-science.ca`. See also `ckanr_setup` and `get_default_url`.
- **...** Curl args passed on to `POST` (optional)

#### Examples

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

Update a resource’s metadata

Description

Update a resource’s metadata

Usage

resource_patch(x, id, 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 (required)

key  
A privileged CKAN API key, Default: your key set with ckanr_setup

url  
Base url to use. Default: http://data.techno-science.ca. See also ckanr_setup and get_default_url.

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 POST (optional)
Examples

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

# Get a resource
res <- resource_show("b85948b6-f9ea-4392-805e-00511d6cf6c6")
res$description

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

## End(Not run)
```

Description

Search for resources.

Usage

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

Arguments

- **q**: Query terms. It is a string of the form `field:term` or a list of strings, each of
  the same form. Within each string, `field` is a field or extra field on the Resource
  domain object. If `field` is hash, then an attempt is made to match the `term` as a
  *prefix* of the Resource.hash field. If `field` 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: `http://data.techno-science.ca`. See also `ckanr_setup` and
  `get_default_url`. 
resource_show

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 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')

## End(Not run)

---

resource_show Show a resource.

Description

Show a resource.

Usage

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

Arguments

id (character) Resource identifier.
url Base url to use. Default: http://data.techno-science.ca. See also ckanr_setup and get_default_url.

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 POST (optional)

Examples

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

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

# then create a resource
resource_update

Update a resource’s file attachment

Description

This function will only update a resource’s file attachment and the metadata key "last_updated". Other metadata, such as name or description, are not updated.

The new file must exist on a local path. R objects 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

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

Arguments

- id (character) Resource ID to update (required)
- path (character) Local path of the file to upload (required)
- key A privileged CKAN API key. Default: your key set with `ckanr_setup`
- url Base url to use. Default: `http://data.techno-science.ca`. See also `ckanr_setup` and `get_default_url`
- 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 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

```r
## Not run:
ckanr_setup[url = "http://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("newpackage3"))
(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)

####
## 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",
```
revision_list

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

Description

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

Usage

revision_list(url = get_default_url(), as = "list", ...)

Arguments

- **url**  
  Base url to use. Default: http://data.techno-science.ca. See also `ckanr_setup` and `get_default_url`.
- **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 `POST` (optional)

Examples

```r
## 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
Comes from the links at http://ckan.org/instances

There were a number of other URLs for CKAN instances in the CKAN URL above, but some sites are now gone completely, or if they do exist, I can’t figure out how to get access to the CKAN API on their instance.

Examples

```r
## Not run:
servers()
ckan_info("http://data.gov.au/")
ckan_info(servers()[5])

# what version is each CKAN server running
out <- lapply(servers()[1:6], ckan_info)
vapply(out, "[[", ",", "ckan_version")

## End(Not run)
```

tag_create
Create a tag

Description
Create a tag

Usage
tag_create(name, vocabulary_id, key = get_default_key(), url = get_default_url(), as = "list", ...)
**tag_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'
- **key** A privileged CKAN API key, Default: your key set with ckanr_setup
- **url** Base url to use. Default: http://data.techno-science.ca. See also ckanr_setup and get_default_url.
- **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 POST (optional)

---

**tag_list**

*List tags.*

**Description**

List tags.

**Usage**

```
tag_list(query = NULL, vocabulary_id = NULL, all_fields = FALSE, url = get_default_url(), 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: http://data.techno-science.ca. See also ckanr_setup and get_default_url.
- **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 POST (optional)
Examples

```r
## 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 List tags.

tag_search

Description
List tags.

Usage
tag_search(query = NULL, vocabulary_id = NULL, offset = 0, limit = 31,
url = get_default_url(), 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 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: `http://data.techno-science.ca`. See also ckanr_setup and get_default_url.
- `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 POST (optional)
**Examples**

```r
tag_search(query = 'ta')
```

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

## End(Not run)

---

**tag_show**  
*Show a tag.*

**Description**

Show a tag.

**Usage**

```r
tag_show(id, include_datasets = FALSE, url = get_default_url(), 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: `http://data.techno-science.ca`. See also `ckanr_setup` and `get_default_url`.
- `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 `POST` (optional)

**Examples**

```r
## 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
```
user_activity_list  

**Return a list of a user’s activities**

### Description

Return a list of a user’s activities

### Usage

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

### Arguments

- **id** (character) User identifier.
- **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: `http://data.techno-science.ca`. See also `ckanr_setup` and `get_default_url`.
- **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 `POST` (optional)

### Examples

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

# list package activity
user_activity_list('sckottie')

# input a ckan_user object
(x <- user_show('sckottie'))
user_activity_list(x)

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

## End(Not run)
user_followee_count

Return a user's follower count

Description

Return a user's follower count

Usage

user_followee_count(id, url = get_default_url(), as = "list", ...)

Arguments

id (character) User identifier.
url Base url to use. Default: http://data.techno-science.ca. See also ckanr_setup and get_default_url.
as (character) One of list (default), table, or json. Parsing with table option uses jsonlite::fromJSON(..., simplifyDataFrame = TRUE), which attempts to parse data to dataframe's when possible, so the result can vary from a vector, list or data.frame. (required)
...

Examples

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

# list package activity
user_followee_count('sckottie')

# input a ckan_user object
(x <- user_show('sckottie'))
user_followee_count(x)

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

## End(Not run)
```
user_followee_list  
Return a user’s follower count

Description

Return a user’s follower count

Usage

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

Arguments

id  (character) User identifier.
url  Base url to use. Default: http://data.techno-science.ca. See also ckanr_setup and get_default_url.
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 POST (optional)

Examples

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

# list package activity
user_followee_list('skottie')

# input a ckan_user object
(x <- user_show('skottie'))
user_followee_list(x)

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

## End(Not run)
user_follower_count

Return a a user’s follower count

Description

Return a a user’s follower count

Usage

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

Arguments

id (character) User identifier.
url Base url to use. Default: http://data.techno-science.ca. See also ckanr_setup and get_default_url.
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 POST (optional)

Examples

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

# list package activity
user_follower_count('sckottie')

# input a ckan_user object
(x <- user_show('sckottie'))
user_follower_count(x)

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

## End(Not run)
user_list  

*Return a list of the site’s user accounts.*

**Description**

Return a list of the site’s user accounts.

**Usage**

```r
user_list(q = NULL, order_by = NULL, url = get_default_url(),
  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: `http://data.techno-science.ca`. See also ckanr_setup and get_default_url.
- `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 POST (optional)

**Examples**

```r
## 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

Show a user.

Description

Show a user.

Usage

user_show(id, user_obj = NULL, include_datasets = FALSE, include_num_followers = FALSE, url = get_default_url(), 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** (character) Base url to use. Default: http://data.techno-science.ca. See also ckanr_setup and get_default_url.
- **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 POST (optional)

Examples

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

# show user
user_show('sckottie')

# include datasets
user_show('sckottie', include_datasets = TRUE)

# include datasets
user_show('sckottie', include_num_followers = TRUE)

## End(Not run)
```
Index

*Topic package
  ckanr-package, 3
  as.ckan_group, 4
  as.ckan_organization, 5
  as.ckan_package, 6
  as.ckan_related, 6
  as.ckan_resource, 7
  as.ckan_tag, 8
  as.ckan_user, 9
  changes, 9
  ckan_classes, 12
  ckan_info, 13
  ckanr (ckanr-package), 3
  ckanr-package, 3
  ckanr_setup, 10
  ckanr_settings, 3, 10, 11, 11, 13–19, 22–32, 34–53, 55, 57–65
  dashboard_activity_list, 14
  dashboard_count, 15
  ds_create, 4, 16
  ds_create_dataset, 4, 17
  ds_search, 4, 18
  ds_search_sql, 4, 19
  fetch, 20

GET, 13, 20
get_default_key, 3, 11
get_default_key (ckanr_settings), 10
get_default_url, 10, 11, 13–19, 22–32, 34–53, 55, 57–65
get_default_url (ckanr_settings), 10
get_test_behaviour (ckanr_settings), 10
get_test_did, 11
get_test_did (ckanr_settings), 10
get_test_gid, 11
get_test_gid (ckanr_settings), 10
get_test_key, 11
get_test_key (ckanr_settings), 10
get_test_oid (ckanr_settings), 10
get_test_oid (ckanr_settings), 10
get_test_rid, 11
get_test_rid (ckanr_settings), 10
get_test_url, 11
get_test_url (ckanr_settings), 10
get_test_url (ckanr_settings), 10
group_create, 21
group_delete, 23
group_list, 3, 23
group_patch, 24
group_show, 4, 25
group_update, 26

is.ckan_group (as.ckan_group), 4
is.ckan_organization (as.ckan_organization), 5
is.ckan_package (as.ckan_package), 6
is.ckan_related (as.ckan_related), 6
is.ckan_resource (as.ckan_resource), 7
is.ckan_tag (as.ckan_tag), 8
is.ckan_user (as.ckan_user), 9
license_list, 27
organization_create, 3, 28
organization_list, 3, 29
organization_show, 3, 5, 30
package_activity_list, 31
package_create, 3, 32
package_delete, 34
package_list, 35
package_list_current, 36
package_patch, 37
package_revision_list, 38
package_search, 39, 59
package_show, 6, 40
package_update, 41
ping, 42
POST, 10, 14–19, 22–32, 34–50, 52, 53, 55, 57–65
read.csv, 4
related_create, 43
related_delete, 44
related_list, 45
related_show, 7, 45
related_update, 46
resource_create, 3, 47
resource_delete, 49
resource_patch, 50
resource_search, 51
resource_show, 7, 52
resource_update, 53
revision_list, 55
servers, 4, 56
tag_create, 56
tag_list, 3, 57
tag_search, 58
tag_show, 8, 59
user_activity_list, 60
user_followee_count, 61
user_followee_list, 62
user_follower_count, 63
user_list, 3, 64
user_show, 9, 65