

# API CLIENTE

## REPOSITORIO DE DATOS

(Compatible con JRO-DB – 2.9.2)

**Versión de guía: 2.9.2.0**

# ÍNDICE

<b>1. INSTALACION Y REQUISITOS PREVIOS .....</b>	<b>1</b>
<b>2. FUNCIONES DISPONIBLES.....</b>	<b>1</b>
<b>2.1. FUNCIÓN: show .....</b>	<b>2</b>
<b>2.2. FUNCIÓN: search .....</b>	<b>3</b>
<b>2.3. FUNCIÓN: create .....</b>	<b>5</b>
<b>2.4. FUNCIÓN: patch .....</b>	<b>10</b>
<b>2.5. FUNCIÓN: delete.....</b>	<b>13</b>
<b>2.6. FUNCIÓN: download.....</b>	<b>14</b>
<b>2.7. FUNCIÓN: action .....</b>	<b>15</b>
<b>3. EJEMPLOS.....</b>	<b>15</b>

## ***ESTA GUÍA SE ACTUALIZARÁ DEPENDIENDO DE NUEVOS AVANCES***

### **¿Cómo reportar algún problema de esta guía?**

Envíe un correo a [eynilupu@igp.gob.pe](mailto:eynilupu@igp.gob.pe) detallando los siguientes pasos:

- 1) Correo electrónico para contactarlo.
- 2) Descripción del problema.
- 3) ¿En qué paso o sección encontró el problema?
- 4) ¿Cuál era el resultado que usted esperaba?

## 1. INSTALACION Y REQUISITOS PREVIOS

1.1. Tener **pip** (python 2) o **pip3** (python 3) instalado

1.2. Instalar la API Cliente

**OBS:** Esta API Cliente está disponible para: Windows - Unix - macOS.

a) En Python 2:

```
pip install -e git+http://intranet.igp.gob.pe:8082/DATABASES/ckanext-jro/api-cliente#egg=jrodb
```

b) En Python 3:

```
pip3 install -e git+http://intranet.igp.gob.pe:8082/DATABASES/ckanext-jro/api-cliente#egg=jrodb
```

## 2. FUNCIONES DISPONIBLES

**OBS:** Usar la función **help()** o el atributo **.\_\_doc\_\_** para la documentación.

Ejemplo 1:

```
from jrodb import Api
help(Api)
```

Ejemplo 2:

```
from jrodb import Api
print(Api.__doc__)
```

Actualmente se dispone de las siguientes funciones para consumirlas:

- |          |            |
|----------|------------|
| ❖ show   | ❖ delete   |
| ❖ search | ❖ download |
| ❖ create | ❖ action   |
| ❖ patch  |            |

¿Cómo consumir alguna operación?

Ejemplo 1)

```
from jrodb import Api

with Api('<host_url>', Authorization='<token_user>', secure=False) as <access_name>:
    ... function 1 ...
    ... function 2 ...
    ... function N ...
```

Ejemplo 2)

```
from jrodb import Api

<access_name> = Api('<host_url>', Authorization='<token_user>', secure=False)
... function 1 ...
... function 2 ...
... function N ...
<access_name>.ckan.close()
```

**OBS:** En algunos casos es necesario una **autorización** - “Authorization”

**OBS:** Si no existe algún inconveniente con el certificado SSL, omitir el parámetro “secure”

## 2.1. FUNCIÓN: **show**

Función personalizada para una búsqueda en específico.

### **Consideraciones:**

*En ciertos casos se necesita tener los permisos necesarios y usar su Api Token.  
Algunas acciones sólo las podrá realizar un usuario sysadmin.*

### Parámetros requeridos:

- **type\_option** :: <class 'str'>
  - Opciones disponibles:  
**dataset, resource, project, collaborator, member, vocabulary, tag, user y job**
- **id** :: <class 'str'>
  - Criterio de búsqueda por:
    - :: **dataset**: id o name del conjunto de datos
    - :: **resource**: id del recurso
    - :: **project**: id o name del proyecto
    - :: **collaborator**: id o name del usuario con rol colaborador
    - :: **member**: id o name del usuario con rol de miembro
    - :: **vocabulary**: id o name del vocabulario
    - :: **tag**: id o name de la etiqueta
    - :: **user**: id o name del usuario
    - :: **job**: id del trabajo en segundo plano

### Parámetros opcionales:

- Criterio de búsqueda por:
  - :: **project**:
    - **include\_datasets** :: <class 'bool'>  
Incluir los conjuntos de datos del proyecto.
  - :: **collaborator**:
    - **capacity** :: <class 'str'>  
Rol del colaborador: **admin, editor** o **member**
  - :: **member**:
    - **permission** :: <class 'str'>  
Permisos asociados:  
**admin: admin**  
**editor: create\_dataset, update\_dataset o delete\_dataset**  
**member: read**
    - **include\_dataset\_count** :: <class 'bool'>  
Conteo de los conjuntos de datos asociados.
  - :: **tag**:
    - **vocabulary\_id** :: <class 'str'>  
id o name del vocabulario asociado al **tag**.
    - **include\_datasets** :: <class 'bool'>  
Incluir una lista truncada de los conjuntos de datos asociados.
  - :: **user**:
    - **user\_obj** :: <class 'user dict'>  
Diccionario de usuario del usuario a consultar
    - **include\_datasets** :: <class 'bool'>  
Incluir una lista truncada de los conjuntos de datos asociados.  
*(Visible por el mismo usuario o sysadmin)*
    - **include\_num\_followers** :: <class 'bool'>  
Conteo de seguidores del usuario.
    - **include\_password\_hash** :: <class 'bool'>  
Incluir el hash de la contraseña almacenada. *(Visible para un usuario sysadmin)*

- `include_plugin_extras :: <class 'bool'>`  
Incluir los plugins extras. *(Visible para un usuario sysadmin)*

### Estructura:

```
<access_name>.show(type_option=<class 'str'>, id=<class 'str'>,
param_1=<class 'param_1'>, ...)
```

## 2.2. FUNCIÓN: **search**

Función personalizada para búsquedas que satisfagan algún criterio.

### Consideraciones:

*En ciertos casos se necesita tener los permisos necesarios y usar su Api Token.  
Algunas acciones sólo las podrá realizar un usuario sysadmin.*

### Parámetros requeridos:

- `type_option :: <class 'str'>`
  - Opciones disponibles:  
*dataset, resource y tag*
- `query :: <class 'dict'>`
  - Busca coincidencia proporcionando múltiples parámetros y sus valores:
  - Parámetros que se pueden incluir en:  
`:: dataset:`
    - `dataset_start_date :: <class 'str'>`  
Fecha mínima de los recursos.
    - `dataset_end_date :: <class 'str'>`  
Fecha máxima de los recursos.
    - `num_resources :: <class 'int'>`  
Número de recursos.
    - `author :: <class 'str'>`  
Autor
    - `autor_email :: <class 'str'>`  
Correo electrónico del autor.
    - `data_type :: <class 'str'>`  
Tipo de data.  
**OBS:** Valores sugeridos: *spectra, SNR y power*
    - `instrument_name :: <class 'str'>`  
Nombre del instrumento.
    - `voc_instrument_type :: <class 'str'>`  
Tipo de instrumento.  
**OBS:** Valores: Consulte el vocabulario -> *instrument\_type*  

```
<access_name>.show(type_option='vocabulary', id='instrument_type')
```
    - `voc_station_name :: <class 'str'>`  
Nombre de la estación.  
**OBS:** Valores: Consulte el vocabulario -> *station\_name*  

```
<access_name>.show(type_option='vocabulary', id='station_name')
```
    - `license_id :: <class 'str'>`  
*id* de la licencia.  
**OBS:** Valores disponibles, consulta la función *action*  

```
<access_name>.action('license_list')
```
    - `license_title :: <class 'str'>`  
Título de la licencia.
    - `maintainer :: <class 'str'>`  
Encargado de administrar.

```
- maintainer_email :: <class 'str'>
    Correo electrónico del encargado de administrar.
- name :: <class 'str'>
    Nombre.
- title :: <class 'str'>
    Título
- notes :: <class 'str'>
    Nota o descripción
- organization :: <class 'str'>
    name del proyecto asociado.

:: resource:
- file_date_min :: <class 'str'>
    Fecha mínima
- file_date_max :: <class 'str'>
    Fecha máxima
- format :: <class 'str'>
    Formato
- voc_file_type :: <class 'str'>
    Tipo
    OBS: Valores disponibles -> (Consulte el vocabulario file_type)
    <access_name>.show(type_option='vocabulary', id='file_type')
- size :: <class 'int'>
    Tamaño en bytes.
- name :: <class 'str'>
    Nombre
- description :: <class 'str'>
    Nota o descripción

:: tag:
- search :: <class 'list'>
    Coincidencias en el nombre de la etiqueta.
```

#### Parámetros opcionales:

- Criterio de búsqueda por:

:: **dataset**:

```
- q :: <class 'str'>
    Busca coincidencia en los parámetros.
- sort :: <class 'str'>
    Clasificación de los resultados: asc, desc
    EJEMPLO: sort='<param_1> asc, <param_2> desc'
- rows :: <class 'int'>
    Limite los datasets devueltos (no altera el campo count), no
    considera el valor de 0 - DEFAULT(10).
- start :: <class 'int'>
    Omite los n primero(s) datasets (no altera el campo count).
- facet :: <class 'bool'>
    Adiciona los campos facets y search facets
- facet_mincount :: <class 'int'>
    Conteo mínimo que se adicionará al campo facets.
    OBS: Debe habilitar el parámetro facet.
- facet_limit :: <class 'int'>
    Limite los parámetros incluidos en los campos facets y
    search facets, no considera el valor de 0 - DEFAULT(50).
    OBS: Debe habilitar el parámetro facet.
```

- **facet\_field** :: <class 'list of strings'>. Incluye los parámetros de la lista en los campos facets y search\_facets.  
**OBS:** Debe habilitar el parámetro **facet**.
- **include\_drafts** :: <class 'list of strings'> Adiciona los datasets con el estado draft. (Visible para un usuario sysadmin)

:: resource:

- **offset** :: <class 'int'> Omite los n primero(s) recursos (no altera el campo count).
- **limit** :: <class 'int'> Limite los recursos devueltos (no altera el campo count).

:: tag:

- **vocabulary\_id** :: <class 'str'> id o name del vocabulario asociado a los tags.
- **limit** :: <class 'int'> Limite los tags devueltos (no altera el campo count).
- **offset** :: <class 'int'> Omite los n primero(s) tags (no altera el campo count).

#### Estructura:

```
<access_name>.search(type_option=<class 'str'>, query=<class 'dict'>, param_1=<class 'param_1'>, ...)
```

## 2.3. FUNCIÓN: **create**

Función personalizada para crear.

#### **Consideraciones:**

Debe tener los permisos necesarios y usar su Api Token.  
Algunas acciones sólo las podrá realizar el usuario sysadmin.

#### Parámetros requeridos:

- **type\_option** :: <class 'str'>
    - Opciones disponibles:  
**resource, dataset, project, member, collaborator, vocabulary, tag, user y views**
  - Criterio de creación por:
- :: resource: (dependerá del parámetro upload o url)

**OBS:** Se debe elegir solo uno, upload o url

```
- upload ::
  :: <class 'str'>
  >> Ruta del folder de los archivos.
  >> Ruta del archivo.

  :: <class 'List of str'>
  >> Lista de las rutas de los archivos.

- url ::
  :: <class 'str'>
  >> Ruta url asociada al recurso.
  >> Ruta del folder a asociar.

  :: <class 'List of str'>
  >> Lista de las urls o folders asociados al recurso.
```



- **package\_id** :: <class 'str'>  
id o name del dataset al que se le adicionará el archivo.
  - **name** ::  
 :: <class 'str'>  
 Nombre asociado al recurso  
 :: <class 'list of str'>  
 Lista de los nombres asociados a los recursos.
  - **file\_date** ::  
 :: <class 'str'>  
 Fecha asociada al recurso  
 :: <class 'list of str'>  
 Lista de las fechas asociadas a los recursos.
  - **file\_type** ::  
 :: <class 'str'>  
 Tipo del recurso asociado  
 :: <class 'list of str'>  
 Lista de los tipos de recursos asociados.
- OBS:** Valores disponibles -> (Consulte el vocabulario *file\_type*)

```
<access_name>.show(type_option='vocabulary', id='file_type')
```

*(OBS) Las listas deben tener el mismo tamaño y orden entre ellas.*

*(OBS) Para el caso de múltiples recursos a crearse; si los valores de los parámetros son repetidos, se puede simplificar a un único valor.*

:: dataset:

- **name** :: <class 'str'>  
 Nombre del dataset, debe ser de 02 a 100 caracteres y ser alfanumérico en minúscula.  
**OBS:** Se acepta '-' y '\_'
- **owner\_org** :: <class 'str'>  
id del proyecto al que pertenecerá.
- **voc\_instrument\_type** :: <class 'str'>  
 Tipo de instrumento.  
**OBS:** Valores: Consulte el vocabulario -> *instrument\_type*

```
<access_name>.show(type_option='vocabulary', id='instrument_type')
```

- **data\_type** :: <class 'str'>  
 Tipo de data.  
**OBS:** Valores sugeridos: *spectra*, *SNR* y *power*

:: project:

- **name** :: <class 'str'>  
 Nombre del proyecto, debe ser de 02 a 100 caracteres y ser alfanumérico en minúscula.  
**OBS:** Se acepta '-' y '\_'

:: member:

- **id** :: <class 'str'>  
id o name del proyecto asociado.
- **username** :: <class 'str'>  
id o name del usuario.
- **role** :: <class 'str'>  
 Rol que tendrá el usuario en el proyecto.  
 Rol del miembro: **admin**, **editor** o **member**

```
:: collaborator:
- id :: <class 'str'>
  id o name del dataset.
- user_id :: <class 'str'>
  id o name del usuario.
- capacity :: <class 'str'>
  Rol que tendrá el colaborador en el dataset.
  Rol del colaborador: admin, editor o member

:: tag:
(Solo para un usuario sysadmin)
- name :: <class 'str'>
  Nombre del tag, debe ser de 02 a 100 caracteres y ser alfanumérico
  en minúscula.
  OBS: Se acepta '-' y '_'
- vocabulary_id :: <class 'str'>
  id del vocabulario contenedor del nuevo tag.

:: vocabulary:
(Solo para un usuario sysadmin)
- name :: <class 'str'>
  Nombre del vocabulario, debe ser de 02 a 100 caracteres y ser
  alfanumérico en minúscula.
  OBS: Se acepta '-' y '_'

:: user:
(Solo para un usuario sysadmin)
- name :: <class 'str'>
  Nombre del usuario, debe ser de 02 a 100 caracteres y ser
  alfanumérico en minúscula.
  OBS: Se acepta '-' y '_'
- email :: <class 'str'>
  Correo electrónico del usuario.
- password :: <class 'str'>
  Contraseña del nuevo usuario, debe ser mayor a 04 caracteres.

:: views:
- select :: <class 'str'>
  Opciones disponibles:
  :: dataset
  - package :: <class 'dict'>
    La metadata completa del dataset a crear los views
    correspondientes de sus recursos. (únicamente se
    crearán las vistas de los que faltan)
  :: resource
  - resource :: <class 'dict'>
    La metadata completa del resource a crear los views
    correspondientes. (si no necesita, no hará ningún
    cambio)
```

**OBS:** Para obtener la metadata completa, use esta API:

```
<access_name>.show(type_option='resource or dataset', id='#####')
```

Al momento de crear recursos, podemos emplear este mismo resultado para crear las vistas, si empleamos la opción de dataset

```
<access_name>.create(type_option='resource',...,...)[ 'package' ]
```

### Parámetros opcionales:

- Criterio de creación por:

:: resource:

- **others** ::

:: <class 'str' or 'list of strings'>

Campos adicionales asociado al recurso, debe tener la siguiente estructura:

o key1:value1

o [key1:value1, key2:value2]

:: <class 'tuple of strings or tuple of list'>

Campos adicionales asociados a todos los recursos, deben tener la siguiente estructura:

o (key1:value1,)

o ([key1:value1, key2:value2], key3:value3,)

- **format** ::

:: <class 'str'>

Formato asociado al recurso

:: <class 'list of str'>

Lista de los formatos asociados a los recursos.

- **description** ::

:: <class 'str'>

Descripción asociada al recurso

:: <class 'list of str'>

Lista de las descripciones asociadas a los recursos.

- **private** ::

:: <class 'bool'>

Establecer como privado el recurso asociado. DEFAULT(False)

:: <class 'list of bool'>

Establecer como privado los recursos asociados. DEFAULT(False)

- **mimetype** ::

:: <class 'str'>

MIME Type del recurso asociado.

:: <class 'list of str'>

Lista de los MIME Types de los recursos asociados.

- **max\_size** :: <class 'float'> (*únicamente con el parámetro upload*)

Tamaño de los archivos por bloque a subir, en MB - DEFAULT(100)

- **max\_count** :: <class 'int'>

Cantidad de archivos por bloque a subir. MAX(999)-DEFAULT(500).

- **ignore\_repetition** :: <class 'bool'>

Ignorar el error si encuentra un recurso con el mismo nombre - DEFAULT(False).

:: dataset:

- **instrument\_name** :: <class 'str'>

Nombre del instrumento.

- **voc\_station\_name** :: <class 'str'>

Nombre de la estación.

**OBS:** Valores: Consulte el vocabulario -> **station\_name**

`<access_name>.show(type_option='vocabulary', id='station_name')`

- **title** :: <class 'str'>

Título

- **private** :: <class 'bool'>  
Establecer el dataset privado.
- **source** :: <class 'str'>  
url que se adjunte al dataset.
- **author** :: <class 'str'>  
Autor
- **author\_email** :: <class 'str'>  
Correo electrónico del autor.
- **maintainer** :: <class 'str'>  
Encargado de administrar.
- **maintainer\_email** :: <class 'str'>  
Correo electrónico del encargado de administrar.
- **license\_id** :: <class 'str'>  
id de la licencia.  
**OBS:** Valores disponibles, consulta la función **action**

`<access_name>.action('license_list')`

- **notes** :: <class 'str'>  
Nota o descripción.
- **version** :: <class 'str'>  
Versión del Dataset.
- **state** :: <class 'str'>  
Estado del dataset - **DEFAULT(active)**  
**OBS:** Valores disponibles: **draft**, **active** y **deleted**
- **tags** :: <class 'list of dict'>  
Tags libres o de vocabularios que se incluirá al dataset.

**:: project:**

- **title** :: <class 'str'>  
Título del proyecto - **DEFAULT(Se asigna automáticamente)**
- **description** :: <class 'str'>  
Nota o descripción.
- **image\_url** :: <class 'str'>  
URL de la imagen anexada al project.  
**OBS:** Se recomienda realizarlo desde el interfaz
- **state** :: <class 'str'>  
Estado del dataset - **DEFAULT(active)**  
**OBS:** Valores disponibles: **active** y **deleted**
- **users** :: <class 'list of dict'>  
Usuarios existentes que se desea agregar al proyecto.

**:: vocabulary:**

*(Acción únicamente para un usuario sysadmin)*

- **tags** :: <class 'list of dict'>  
Tags que se desea crear y agregar a este vocabulario.

**:: user:**

*(Acción únicamente para un usuario sysadmin)*

- **fullname** :: <class 'str'>  
Nombre completo del nuevo usuario.
- **about** :: <class 'str'>  
Descripción o nota del nuevo usuario.
- **image\_url** :: <class 'str'>  
URL de la imagen anexada al nuevo usuario.  
**OBS:** Se recomienda realizarlo desde el interfaz
- **plugin\_extras** :: <class 'dict'>  
Adicionar características únicas del usuario.

```
:: views:
- De las opciones disponibles, del parámetro select:
  :: dataset y resource
  - create_datastore_views :: <class 'bool'>
    Si el valor es True, al (los) recurso(s) que
    requiera(n) vistas relacionas a la extensión
    datapusher (DataStore), las incluirá. - DEFAULT(False)
```

#### Estructura:

```
<access_name>.create(type_option=<class 'str'>, param_1=<class 'param_1'>, ...)
```

## 2.4. FUNCIÓN: **patch**

Función personalizada para actualizar.

#### **Consideraciones:**

*Debe tener los permisos necesarios y usar su Api Token.*

#### Parámetros requeridos:

- **type\_option** :: <class 'str'>
  - Opciones disponibles:
 

**resource, dataset, project, resource, member y collaborator**
- Criterio de actualización por:
 

```
:: resource:
- id :: <class 'str'>
  id del recurso.
- package_id :: <class 'str'>
  id del dataset.

(OBS) Las Listas deben tener el mismo tamaño y orden entre ellas.
(OBS) Para el caso de múltiples recursos a crearse; si los valores de
los parámetros son repetidos, se puede simplificar a un único valor.

:: dataset:
- id :: <class 'str'>
  id del dataset.

:: project:
- id :: <class 'str'>
  id del proyecto.

:: member:
- id :: <class 'str'>
  id o name del proyecto asociado.
- username :: <class 'str'>
  id o name del usuario.
- role :: <class 'str'>
  Nuevo rol que tendrá el usuario en el proyecto.
  Rol del miembro: admin, editor o member

:: collaborator:
- id :: <class 'str'>
  id o name del dataset.
- user_id :: <class 'str'>
  id o name del usuario.
- role :: <class 'str'>
  Nuevo rol que tendrá el colaborador en el dataset.
  Rol del colaborador: admin, editor o member
```

### Parámetros opcionales:

- Criterio de actualización por:  
:: resource: (*dependerá del parámetro upload o url*)

**OBS:** Se puede cambiar de parámetro, pero debe elegir solo uno:

- **upload** ::  
:: <class 'str'>  
Nueva ruta del archivo.  
:: <class 'list of str'>  
Lista de las nuevas rutas de los archivos.
- **url** ::  
:: <class 'str'>  
Ruta de la nueva url o folder asociado al recurso.  
:: <class 'list of str'>  
Lista de los nuevos urls o folders asociados al recurso.

- **name** ::  
:: <class 'str'>  
Nuevo nombre asociado al recurso  
:: <class 'list of str'>  
Lista de los nuevos nombres asociados a los recursos.
- **file\_date** ::  
:: <class 'str'>  
Nueva fecha asociada al recurso  
:: <class 'list of str'>  
Lista de las nuevas fechas asociadas a los recursos.
- **file\_type** ::  
:: <class 'str'>  
Nuevo tipo del recurso asociado  
:: <class 'list of str'>  
Lista de los nuevos tipos de recursos asociados.

**OBS:** Valores disponibles -> (Consulte el vocabulario *file\_type*)

```
<access_name>.show(type_option='vocabulary', id='file_type')
```

- **others** ::  
:: <class 'str' or 'list of strings'>  
Nuevos campos adicionales asociado al recurso, debe tener la siguiente estructura:  
o key1:value1  
o [key1:value1, key2:value2]  
:: <class 'tuple of strings or tuple of list'>  
Nuevos campos adicionales asociados a todos los recursos, deben tener la siguiente estructura:  
o (key1:value1,)  
o ([key1:value1, key2:value2], key3:value3,)
- **format** ::  
:: <class 'str'>  
Nuevo formato asociado al recurso  
:: <class 'list of str'>  
Lista de los nuevos formatos asociados a los recursos.
- **description** ::  
:: <class 'str'>  
Nueva descripción asociada al recurso  
:: <class 'list of str'>  
Lista de las nuevas descripciones asociadas a los recursos.

- **private** ::  
 :: <class 'bool'>  
 Establecer como privado el recurso asociado. DEFAULT(*False*)  
 :: <class 'list of bool'>  
 Establecer como privado los recursos asociados. DEFAULT(*False*)
- **mimetype** ::  
 :: <class 'str'>  
 Nuevo MIME Type del recurso asociado.  
 :: <class 'list of str'>  
 Lista de los nuevos MIME Types de los recursos asociados.
- **max\_size** :: <class 'float'> (únicamente con el parámetro upload)  
 Tamaño de los archivos por bloque, en MB - DEFAULT(100)
- **max\_count** :: <class 'int'> (únicamente con el parámetro upload)  
 Cantidad de archivos por bloque. MAX(999)-DEFAULT(500).

:: dataset:

- **instrument\_name** :: <class 'str'>  
 Nuevo nombre del instrumento.
- **voc\_station\_name** :: <class 'str'>  
 Nuevo nombre de la estación.  
**OBS:** Valores: Consulte el vocabulario -> **station\_name**  

<access\_name>.show(type\_option='vocabulary', id='station\_name')
- **title** :: <class 'str'>  
 Nuevo título
- **source** :: <class 'str'>  
 Nueva url adjuntada al dataset.
- **author** :: <class 'str'>  
 Nuevo autor
- **author\_email** :: <class 'str'>  
 Correo electrónico del nuevo autor.
- **maintainer** :: <class 'str'>  
 Nuevo encargado de administrar.
- **maintainer\_email** :: <class 'str'>  
 Correo electrónico del nuevo encargado de administrar.
- **license\_id** :: <class 'str'>  
id de la licencia.  
**OBS:** Valores disponibles, consulta la función **action**  

<access\_name>.action('license\_list')
- **private** :: <class 'bool'>  
 Establecer el dataset privado.
- **notes** :: <class 'str'>  
 Nueva nota o descripción.
- **version** :: <class 'str'>  
 Nueva versión del dataset.
- **state** :: <class 'str'>  
 Nuevo estado del dataset - DEFAULT(*active*)  
**OBS:** Valores disponibles: *draft*, *active* y *deleted*
- **tags** :: <class 'list of dict'>  
 Tags libres o de vocabularios para modificar.
- **data\_type** :: <class 'str'>  
 Nuevo tipo de data.  
**OBS:** Valores sugeridos: *spectra*, *SNR* y *power*
- **owner\_org** :: <class 'str'>  
id del proyecto a modificar.

- **voc\_instrument\_type** :: <class 'str'>  
Nuevo tipo de instrumento.  
**OBS:** Valores: Consulte el vocabulario -> **instrument\_type**  

`<access_name>.show(type_option='vocabulary', id='instrument_type')`
  - **dataset\_start\_date** :: <class 'str'>  
Nueva fecha de inicio del dataset - Representa la primera fecha de sus recursos.
  - **dataset\_end\_date** :: <class 'str'>  
Nueva fecha final del dataset - Representa la última fecha de sus recursos.
- :: project:**
- **title** :: <class 'str'>  
Nuevo título del proyecto.
  - **description** :: <class 'str'>  
Nueva nota o descripción.
  - **users** :: <class 'list of dict'>  
Nuevos usuarios a modificar.

#### Estructura:

`<access_name>.patch(type_option=<class 'str'>, param_1=<class 'param_1'>, ...)`

## 2.5. FUNCIÓN: **delete**

Función personalizada para eliminar y/o purgar.

#### **Consideraciones:**

*Debe tener los permisos necesarios y usar su Api Token.  
Algunas acciones sólo las podrá realizar el usuario sysadmin.*

#### Parámetros requeridos:

- **type\_option** :: <class 'str'>
  - Opciones disponibles:  
**resource, dataset, project, vocabulary, tag y user**
- **select** :: <class 'str'>
  - Opciones disponibles:  
**delete y purge**
  - La opción **purge** solo está disponible para **resource, dataset y project**:  
(OBS) Para el caso de 'dataset' y 'project' (purgar) es eliminar toda información sin poder recuperarla en un futuro.  
(OBS) Para el caso de 'resource' (purgar) es eliminar el archivo anexo al recurso, en el caso existiera tal archivo.
- **id** :: <class 'str'> (para **resource** también puede ser <'list of str'>)  
Criterio de eliminación por:  
 :: **resource**:  
   >> id del recurso.  
   >> Lista de los ids de los recursos. En este caso, también se necesita el parámetro **package\_id**  
   :: **dataset**: id o name del conjunto de datos.  
   :: **project**: id o name del proyecto.  
   :: **vocabulary**: id o name del vocabulario. (Solo por un usuario **sysadmin**)  
   :: **tag**: id de la etiqueta. (Solo por un usuario **sysadmin**)  
   :: **user**: id o name del usuario. (Solo por un usuario **sysadmin**)



### Estructura:

```
<access_name>.delete(type_option=<class 'str'>, select=<class 'str'>, id=<class 'str'>)
```

## 2.6. FUNCIÓN: **download**

Función personalizada para la descarga de los archivos de un(os) dataset(s) que comparten los mismos filtros a nivel de recursos.

### **Consideraciones:**

- Para la descarga de algunos recursos debe tener permiso - acceso al dataset y al recurso.
- En ciertos casos necesitará el Api Token.
- Dentro de la ruta señalada, se generará un archivo .json acerca de los recursos descargados.

### Parámetros requeridos:

- **id** :: <class 'str' or 'list'>
  - id o name de un(os) dataset(s), debe tener la siguiente estructura:
    - dataset1
    - [dataset1, dataset2]

### Parámetros opcionales:

- **processes** :: <class 'int'>
  - Número de procesos de trabajo - *DEFAULT(1)*.
- **path** :: <class 'str'>
  - Ruta para la descarga de los archivos - *DEFAULT(Ruta por defecto)*.
  - Sistemas Operativos disponibles: *Linux, Windows, macOS*
- **page** :: <class 'int'>
  - Número de la página que desea descargar; consulte la interfaz.
- **file\_date\_min** :: <class 'str'>
  - Fecha mínima.
- **file\_date\_max** :: <class 'str'>
  - Fecha máxima.
- **description** :: <class 'str'>
  - Nota o descripción de los recursos.
- **private** :: <class 'bool'>
  - Visibilidad de los recursos.
- **format** :: <class 'str'>
  - Formato de los recursos.
- **mimetype** :: <class 'str'>
  - MIME Type de los archivos.
- **name** :: <class 'str'>
  - Nombre de los archivos.
- **size** :: <class 'int'>
  - Tamaño en bytes de los archivos.
- **voc\_file\_type** :: <class 'str'>
  - Tipo de los recursos.
  - OBS:** Valores disponibles -> (Consulte el vocabulario *file\_type*)

```
<access_name>.show(type_option='vocabulary', id='file_type')
```

- **datastore\_active** :: <class 'bool'>
  - Seleccionar recursos que solo tengan información en DataStore.

- **order\_by** :: <class 'str'>
  - Orden de los archivos, sin considerar el parámetro *page*.
- **others\_key1 / others\_keyN** :: <class 'str' or 'list of strings'>
  - Campos adicionales, deben tener la siguiente estructura:
    - value1
    - [value1, value2]

Estructura:

```
<access_name>.download(id=<class 'str'>, param_1=<class 'param_1'>, ...)
```

## 2.7. FUNCIÓN: **action**

**OBS:** Esta función es para un uso avanzado del repositorio, debe tener precaución al emplear tal función y su respectivo consumo de las APIs

Esta función engloba todas las APIs que dispone CKAN 2.9.2, incluye todas las acciones (get, create, update, patch y delete) y de las extensiones externas e internas adicionadas a CKAN 2.9.2.

Al usar esta función para consumir las APIs debe considerar las configuraciones en su estructura y metadata que se adicionaron y/o modificaron para tener la personalización que requiere el Repositorio del ROJ. Por ello, al consumir cualquier API usando esta función deberá realizar con precaución.

Las APIs disponibles de CKAN 2.9.2. , con su respectiva documentación, las puede ubicar en:

<https://docs.ckan.org/en/2.9/api/index.html#action-api-reference>  
<https://docs.ckan.org/en/2.9/maintaining/datastore.html#the-datastore-api>  
<https://github.com/ckan/ckanext-scheming/blob/master/ckanext/scheming/logic.py>  
<https://github.com/ckan/ckanext-harvest/tree/master/ckanext/harvest/logic/action>

(recuerde tener en consideración las configuraciones y la metadata adicionada para el funcionamiento del Repositorio del ROJ)

Adicionalmente, se adicionaron estas 02 APIs de la extensión **ckanext-jroconfig**:

### 1) **resources\_search**

```
'''Searches for resources satisfying a given search criteria.
- query :: <class 'dict'>
.....
- file_date_min :: <class 'str'>
- file_date_max :: <class 'str'>
- format :: <class 'str'>
- voc_file_type :: <class 'str'>
- size :: <class 'int'>
- name :: <class 'str'>
- description :: <class 'str'>
.....
- order_by :: <class 'str'>
- limit :: <class 'int'>
- offset :: <class 'int'>
...'''
```

## 2) `url_resources`

*'''Return the url of resources in a dataset*

```
- param id :: <class 'str'>
    "id" or "name" of the dataset
.....
- name :: <class 'str' or 'list of str'>
- voc_file_type :: <class 'str' or 'list of str'>
- file_date_min :: <class 'str'>
- file_date_max :: <class 'str'>
- format :: <class 'str' or 'list of str'>
- private :: <class 'boolean' or 'list of boolean'>
- size :: <class 'int' or 'list of int'>
- description :: <class 'str' or 'list of str'>
- mimetype :: <class 'str' or 'list of str'>
- datastore_active :: <class 'boolean' or 'list of boolean'>
- page :: <class 'int'>
- order_by :: <class 'str'>
---
- others_key1 :: <class 'str' or 'list of str'>
- others_keyN :: <class 'str' or 'list of str'>
.....
'''
```

### Estructura:

```
<access_name>.action(<consuming API>, param_1=<class 'param_1'>, ...)
```

### 3. EJEMPLOS

#### FUNCIÓN: **show**

```
from jrodb import Api

with Api('http://10.10.20.128:5000', Authorization='#####') as access:

    access.show(type_option='resource', id='#####')
```

```
from jrodb import Api

with Api('http://localhost', Authorization='#####') as access:

    access.show(type_option='dataset', id='dataset_name')
```

```
from jrodb import Api

with Api('http://localhost', Authorization='#####') as access:

    access.show(type_option='project', id='project_name', include_datasets=True)
```

Si no se declara el parámetro **vocabulary\_id** se asume que el **tag** no pertenece a un vocabulario.

```
from jrodb import Api

with Api('http://10.10.20.128:5000', Authorization='#####') as access:

    access.show(type_option='tag', id='tag_name', vocabulary_id='#####')
```

```
from jrodb import Api

with Api('http://localhost', Authorization='#####') as access:

    access.show(type_option='collaborator', id='collaborator_name', capacity='admin')
```

```
from jrodb import Api

with Api('http://localhost', Authorization='#####') as access:

    access.show(type_option='member', id='member_name', permission='read')
```

## FUNCIÓN: **search**

```
from jrodb import Api

with Api('http://10.10.20.128:5000', Authorization='#####') as access:

    access.search(type_option='dataset', query={'dataset_start_date': '2021-03-01'})
```

```
from jrodb import Api

with Api('http://localhost', Authorization='#####') as access:

    access.search(type_option='dataset', query={'dataset_end_date': '2021-03-31', 'voc_station_name': 'Huancayo'}, q='dataset_name', sort='author asc', start=2)
```

```
from jrodb import Api

with Api('http://10.10.20.128:5000', Authorization='#####') as access:

    access.search(type_option='dataset', query={'dataset_start_date': '2021-03-01', 'dataset_end_date': '2021-03-31'}, facet=True, facet_field=['param1', 'param2'], facet_limit=10)
```

```
from jrodb import Api

with Api('http://localhost', Authorization='#####') as access:

    access.search(type_option='resource', query={'file_date_min': '2021-03-15'})
```

```
from jrodb import Api

with Api('http://10.10.20.128:5000', Authorization='#####') as access:

    access.search(type_option='resource', query={'file_date_max': '2021-03-31', 'format': 'PNG'}, limit=5, offset=1)
```

```
from jrodb import Api

with Api('http://10.10.20.128:5000', Authorization='#####') as access:

    access.search(type_option='tag', query={'search': ['tag_name_truncated']})
```

```
from jrodb import Api

with Api('http://localhost', Authorization='#####') as access:

    access.search(type_option='tag', query={'search': ['value1', 'value2']}, vocabulary_id='vocabulary_name', limit=3)
```



## FUNCIÓN: **create**

```
from jrodb import Api

with Api('https://www.igp.gob.pe/observatorios/radio-observatorio-jicamarca/database', Authorization='#####') as access:

    access.create(type_option='resource', package_id='#####', upload='/path/to/file01.png', name='image01.png', file_date='2022-12-01', file_type='Image')
```

```
from jrodb import Api

with Api('http://10.10.110.243:8085/observatorios/radio-observatorio-jicamarca/database', Authorization='#####') as access:

    access.create(type_option='resource', package_id='#####', upload=['/path/to/file01.png', '/path/to/file02.hdf5'], file_date='2022-12-01', file_type=['Image', 'Data'],
        others=(['key1:value1', 'key2:value2', 'key3:value3'], format=['PNG', 'HDF5'], max_size=4, ignore_repetition=True)
```

EXAMPLE FOLDER:

```
root
├── workspace
│   ├── image_01.png
│   ├── ...
│   └── image_NN.png
```

```
from jrodb import Api

with Api('http://10.10.10.78:8085/observatorios/radio-observatorio-jicamarca/database', Authorization='#####') as access:

    access.create(type_option='resource', package_id='#####', upload='/root/workspace', file_date='2021-03-01', file_type='Image', format='PNG',
        description='A Little information', others=(['key1:value1', 'key2:value2', 'key3:value3'], max_size=10, max_count=100)
```

```
from jrodb import Api

with Api('https://www.igp.gob.pe/observatorios/radio-observatorio-jicamarca/database', Authorization='#####') as access:

    access.create(type_option='resource', package_id='#####', url='https://www.igp.gob.pe/file01.png', name='image01.png', file_date='2022-12-01', file_type='Image')
```

```
from jrodb import Api

with Api('http://10.10.110.243:8085/observatorios/radio-observatorio-jicamarca/database', Authorization='#####') as access:

    access.create(type_option='resource', package_id='#####', url=['https://www.igp.gob.pe/file01.png', 'https://www.igp.gob.pe/file02.png'], file_date='2022-12-01',
        file_type='Image', others=(['key1:value1', 'key2:value2'], ['key1:value1', 'key3:value3'], format='PNG', max_count=1, ignore_repetition=True)
```

```
from jrodb import Api

with Api('https://www.igp.gob.pe/observatorios/radio-observatorio-jicamarca/database', Authorization='#####') as access:

    access.create(type_option='resource', package_id='#####', url='minotaur//data/workspace_2022_02', name='Files-February-2022', file_date='2022-02-01', file_type='Image')
```

```
from jrodb import Api

with Api('http://10.10.20.128:5000', Authorization='#####') as access:

    access.create(type_option='resource', package_id='#####', url=['minotaur//data/works_2021_01', 'https://www.igp.gob.pe/file01.txt'], file_date=['2021-01-01', '2020-12-12'],
        file_type='Data', format=['', 'TXT'], others='key1:value1', max_count=2, ignore_repetition=True)
```



```
from jrodb import Api

with Api('http://10.10.20.128:5000', Authorization='#####') as access:

    access.create(type_option='dataset', owner_org='#####', voc_instrument_type='Wind profiler', data_type='SNR')
```

```
from jrodb import Api

with Api('http://localhost', Authorization='#####') as access:

    access.create(type_option='dataset', owner_org='#####', voc_instrument_type='Imagers', data_type='SNR', tags = [{ 'id': '#####', 'name': 'tag_name', 'vocabulary_id': '#####'},
    { 'name': 'new_tag' }], private=True, notes='Dataset notes', license_id='#####')
```

```
from jrodb import Api

with Api('http://localhost', Authorization='#####') as access:

    access.create(type_option='project', name='project_name')
```

```
from jrodb import Api

with Api('http://10.10.20.128:5000', Authorization='#####') as access:

    access.create(type_option='project', name='proyecto_name', description='Project notes', users=[{ 'name': 'user1_name', 'capacity': 'admin' }])
```

```
from jrodb import Api

with Api('http://localhost', Authorization='#####') as access:

    access.create(type_option='member', id='project_name', username='user_name', role='editor')
```

```
from jrodb import Api

with Api('http://10.10.20.128:5000', Authorization='#####') as access:

    access.create(type_option='collaborator', id='dataset_name', user_id='user_name', capacity='member')
```

```
from jrodb import Api

with Api('http://localhost', Authorization='#####') as access:

    access.create(type_option='vocabulary', name='vocabulary_name')
```

```
from jrodb import Api

with Api('http://localhost', Authorization='#####') as access:

    access.create(type_option='vocabulary', name='vocabulary_name', tags=[{ 'name': 'tag_name' }])
```

```
from jrodb import Api

with Api('http://10.10.20.128:5000', Authorization='#####') as access:

    access.create(type_option='tag', name='tag_name', vocabulary_id='#####')
```

```
from jrodb import Api

with Api('http://10.10.20.128:5000', Authorization='#####') as access:

    access.create(type_option='user', name='user_name', email='user@igp.gob.pe', password='*****')
```

```
from jrodb import Api

with Api('http://localhost', Authorization='#####') as access:

    access.create(type_option='user', name='user_name', email='user@igp.gob.pe', password='*****', about='User account', plugin_extras={'extra': {'param1': 'value1'}})
```

```
from jrodb import Api

with Api('http://localhost', Authorization='#####') as access:

    metadata_dataset = access.show(type_option='dataset', id='dataset_name')

    access.create(type_option='views', select='dataset', package = metadata_dataset)
```

```
from jrodb import Api

with Api('http://localhost', Authorization='#####') as access:

    create_res = access.create(type_option='resource', package_id='#####', upload='/path/to/file01.png', name='image01.png', file_date='2022-12-01', file_type='Image')

    access.create(type_option='views', select='dataset', package = create_res['package'])
```

```
from jrodb import Api

with Api('http://localhost', Authorization='#####') as access:

    metadata_resource = access.show(type_option='resource', id='resource_name')

    access.create(type_option='views', select='resource', resource = metadata_resource)
```



## FUNCIÓN: **patch**

```
from jrodb import Api

with Api('https://www.igp.gob.pe/observatorios/radio-observatorio-jicamarca/database', Authorization='#####') as access:

    access.patch(type_option='resource', package_id='#####', id='resource_id', url='https://www.igp.gob.pe/file01.png', file_date='2022-12-01')
```

```
from jrodb import Api

with Api('http://10.10.20.128:5000', Authorization='#####') as access:

    access.patch(type_option='resource', package_id='#####', id=['resource_id_01', 'resource_id_02'], path=['/workspace/to/file01.png', '/workspace/to/file02.png' ], max_size=5)
```

```
from jrodb import Api

with Api('https://www.igp.gob.pe/observatorios/radio-observatorio-jicamarca/database', Authorization='#####') as access:

    access.patch(type_option='resource', package_id='#####', id='resource_id', file_type='Image')
```

```
from jrodb import Api

with Api('http://localhost', Authorization='#####') as access:

    access.patch(type_option='dataset', id='dataset_name', notes='Dataset notes')
```

```
from jrodb import Api

with Api('http://10.10.20.128:5000', Authorization='#####') as access:

    access.patch(type_option='dataset', id='dataset_name', notes='Dataset notes', tags=[{'name': 'tag_name'}, {'id': 'tag_id', 'name': 'tag_name', 'vocabulary_id': 'vocabulary_id'}])
```

```
from jrodb import Api

with Api('http://localhost', Authorization='#####') as access:

    access.patch(type_option='project', id='project_name', description='Project notes')
```

```
from jrodb import Api

with Api('http://localhost', Authorization='#####') as access:

    access.patch(type_option='project', id='project_name', description='Project notes', users=[{'name': 'user1', 'capacity': 'editor'}, {'name': 'user2', 'capacity': 'editor'}])
```

```
from jrodb import Api

with Api('http://localhost', Authorization='#####') as access:

    access.patch(type_option='member', id='project_name', role='admin', username='user_name')
```

```
from jrodb import Api

with Api('http://localhost', Authorization='#####') as access:

    access.patch(type_option='collaborator', id='dataset_name', user_id='user_name', capacity='member')
```

## **FUNCIÓN: delete**

```
from jrodb import Api

with Api('http://10.10.20.128:5000', Authorization='#####') as access:

    access.delete(type_option='resource', select='purge', id='resource_id')
```

```
from jrodb import Api

with Api('http://10.10.20.128:5000', Authorization='#####') as access:

    access.delete(type_option='resource', select='purge', package_id='#####', id=['resource_id_01', 'resource_id_02'] )
```

```
from jrodb import Api

with Api('http://localhost', Authorization='#####') as access:

    access.delete(type_option='dataset', select='purge', id='dataset_name')
```

```
from jrodb import Api

with Api('http://localhost', Authorization='#####') as access:

    access.delete(type_option='project', select='purge', id='project_name')
```

## **FUNCIÓN: download**

```
from jrodb import Api  
  
with Api('http://localhost', Authorization='#####') as access:  
    access.download(id='dataset_id')
```

```
from jrodb import Api  
  
with Api('http://10.10.20.128:5000', Authorization='#####') as access:  
    access.download(id='dataset_id', path='/root/username/workspace', processes=2)
```

```
from jrodb import Api  
  
with Api('http://localhost', Authorization='#####') as access:  
    access.download(id=['dataset_1', 'dataset_2'], path='C:\\Users\\Username\\workspace', file_date_min='2021-03-01', page=3, format='PNG')
```

## **FUNCIÓN: action**

```
from jrodb import Api  
  
with Api('http://localhost', Authorization='#####') as access:  
    access.action('license_list')
```

```
from jrodb import Api  
  
with Api('http://10.10.20.128:5000', Authorization='#####') as access:  
    access.action('vocabulary_list')
```

```
from jrodb import Api  
  
with Api('http://10.10.20.128:5000', Authorization='#####') as access:  
    access.action('status_show')
```