Leica ConX ONE API

Pagination

On endpoints that support pagination there are two variables in the response called <code>next</code> and <code>previous</code>.

Repeating the request using the URL will return the next or previous page of results. If no more pages are available they will contain a null value.

Authentication

CURRENT LOGIN

Manage the current login.

GET /api/one/v1/me/ Get my Profile

Get information about the currently logged in user or device. This will return a JSON object with the following schema:

- · username user name of the current login
- uuid ID of the current login
- · parent account to which the current login belongs
 - type type of account
 - uuid ID of the account
 - $\circ \hspace{0.1in}$ name name of the account
- parent_unit parent unit account to which the current login belongs
 - o uuid unit uuid
 - o name unit name
- parent_project parent project account to which the current login belongs
 - o uuid project uuid
 - o name project name
- parent_company parent company account to which the current login belongs
 - o uuid company uuid
 - o name company name
 - o company_type company type

Example URI

GET https://conx.leica-geosystems.com/api/one/v1/me/

Request Show

Response 200

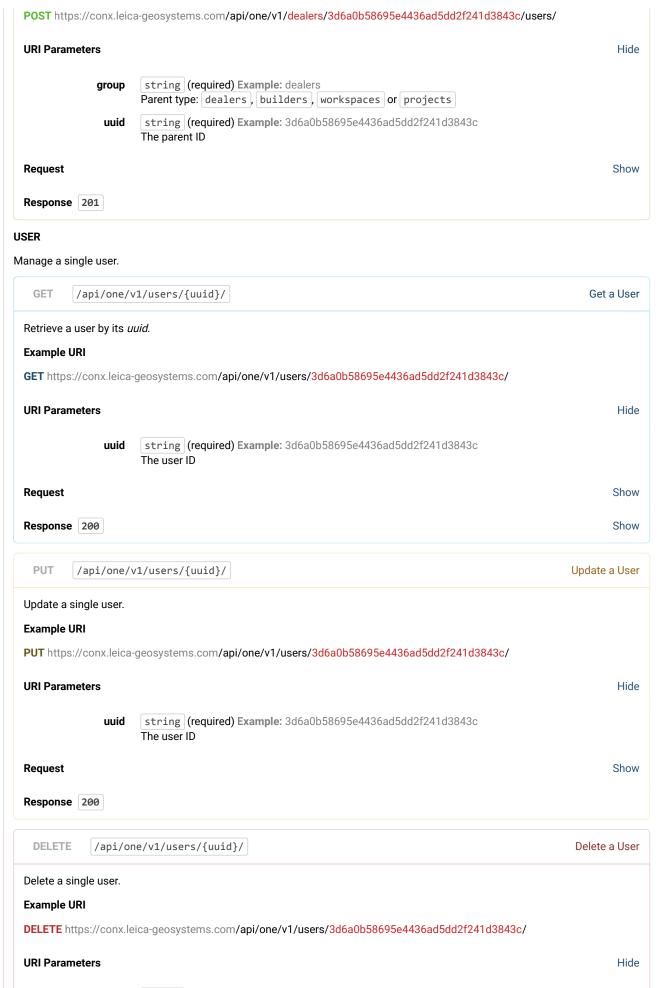
Show

DELETE /api/one/v1/me/

Delete my Login

Delete the user along with its connection. After calling this the user and with the token will be revoked and access to APIs will be denied.

When all connections to a project are removed, all files in the project will be untracked and remain as unmanaged files not visible through any API, even if a new connection is established at a later time. **Example URI** DELETE https://conx.leica-geosystems.com/api/one/v1/me/ Response 204 **MASTER USER LIST** /api/one/v1/master/users/ **Get Users** Get a list of users. **Example URI** GET https://conx.leica-geosystems.com/api/one/v1/master/users/ Request Show Response 200 Show Create New User **POST** /api/one/v1/master/users/ Create a new user. **Example URI** POST https://conx.leica-geosystems.com/api/one/v1/master/users/ Request Show Response 201 **GROUP USER LIST** /api/one/v1/{group}/{uuid}/users/ Get Users **GET** Get a list of users. **Example URI** GET https://conx.leica-geosystems.com/api/one/v1/dealers/3d6a0b58695e4436ad5dd2f241d3843c/users/ **URI Parameters** Hide string (required) Example: dealers group Parent type: dealers, builders, workspaces or projects uuid string (required) Example: 3d6a0b58695e4436ad5dd2f241d3843c The parent ID Request Show Response 200 Show POST /api/one/v1/{group}/{uuid}/users/ Create New User Create a new user. **Example URI**



Integration

INTEGRATING WEB SERVICES

Request authorization to access a Leica ConX project by third-party applications. This authorization should be used when a construction project is managed by Leica ConX as well as a third-party application, and there is a need to synchronize information between both. To be able to use this service, the third-party application must have a *service token* provided by Leica ConX.

POST /api/one/v1/integration/sessions/

Create New Session

Create a new application connection session. The third-party application must provide its service token via the *ServiceToken* request header. It must also provide a JSON object with the following components:

- external_ref the ID of the corresponding project in the application
- project_name the name of the corresponding project in the application
- callback_url application end-point to which authorization is granted
- redirect_url application end-point to which the end-user is returned

Given a valid service token and JSON object, the client should receive a JSON object containing the following components:

- session_id the session ID with which to track this connection session
- url Leica ConX URL where the user should authorize the connection

The application should then re-direct the user agent to Leica ConX as specified in [url].

Granting Authorization

Once in Leica ConX, the user may authorize connection to a project. Specifically, the user should select the corresponding project in Leica ConX against which the connection should be granted.

Note that, to perform these steps in Leica ConX, the user must have an account in Leica ConX and be logged in as well.

If granted, Leica ConX will make a POST request against callback_url , provided by the third-party application, with the same service token via the ServiceToken request header. A URL encoded web form with the following components will be included:

- token project token
- · session_id the session ID with which to track this connection session
- · uuid the ID of the corresponding project in Leica ConX
- project_name The name of the corresponding project in Leica ConX

The third-party application should securely store the token and use it when synchronizing related project data. To acknowledge the connection, the application should respond with HTTP status 200, otherwise the response body should include a JSON object containing the following component:

· error - a text describing the result of the operation

Leica ConX will then re-direct the user agent back to the third-party application as specified in redirect_ur1. A URL encoded web form with the following components will be included:

- session_id the session ID with which to track this connection session
- · result a code indicating the result of this connection session
- status HTTP status from the callback_url response or from Leica ConX
- error description from the callback_url response or from Leica ConX

The result code may be one of the following:

Result Description

| Result | Description |
|--------|---|
| ОК | Connection established |
| CANCEL | End-user aborted connection process |
| CBKERR | An error was returned from callback_url |
| SRVERR | An error ocurred in Leica ConX |

When result is set to CBKERR, the error and status attributes will match the response from callback_url. When set to SRVERR, the attributes will match the condition reported by Leica ConX to the user agent when re-directing back to the third-party application. Otherwise, error and status will be empty.

Example URI

POST https://conx.leica-geosystems.com/api/one/v1/integration/sessions/

Request Show

Response 200 Show

THIRD-PARTY CONNECTION LIST

Authorize access to a Leica ConX project for a third-party application. The request should be made from the Leica ConX web client by a logged in user. It must provide a JSON object with the following components:

- project_uuid the ID of the Leica ConX project to authorize access to
- · session_id the associated connection session ID

Leica ConX will lookup the parameters applicable to the connection session, as identified by session_id, and attempt to authorize the connection against the third-party application. For a description of this mechanism, including the response that should be expected from this end-point, refer to *Granting Authorization*.

When re-directing back to the third-party application, components in the response should be provided by way of a URL encoded web form. Example:

https://example.com/icon?result=CBKERR&status=500&error=

POST /api/one/v1/integration/connections/

Create New Connection

Create a new connections. This will, depending on the locking approach, set restrictions in place for working with the project through the UI

Example URI

POST https://conx.leica-geosystems.com/api/one/v1/integration/connections/

Request Show

Response 201

Show

THIRD-PARTY CONNECTION

Revoke a connection to a third-party site.

DELETE /api/one/v1/integration/connections/{uuid}/

Delete a Connection

Example URI

DELETE https://conx.leica-geosystems.com/api/one/v1/integration/connections/9944b09199c62bcf9418ad846dd0e4bbdfc6e e4b/

URI Parameters Hide

uuid string (required) Example: 9944b09199c62bcf9418ad846dd0e4bbdfc6ee4b
The connection ID

Request

Response 204

LEGACY FEATURE RESTRICTIONS

There are two different approaches to handle restrictions of legacy features when an integration connection is made.

- Pessimistic: where a user is not able to rename project, upload files, delete files, or edit files through the UI as soon as an integration connection is made to the project.
- Optimistic: where the restrictions is set in affect only after a design file is uploaded through the API.

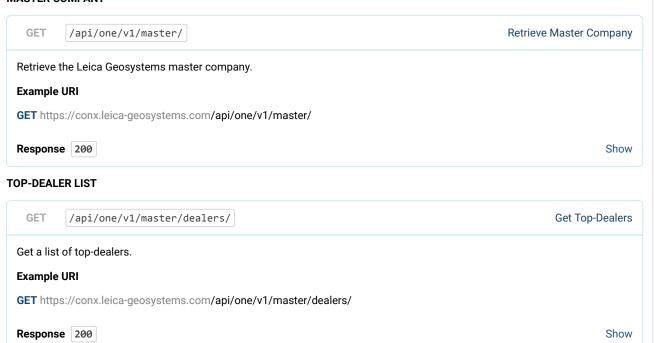
The restrictions are:

- Rename
- Delete
- Upload
- · Create new File
- · Paste file

Company

Companies can be of three types: master, dealer or construction. A master company is not related by another company; a construction company is not related to another company.

MASTER COMPANY



Create a new top-dealer.

Example URI

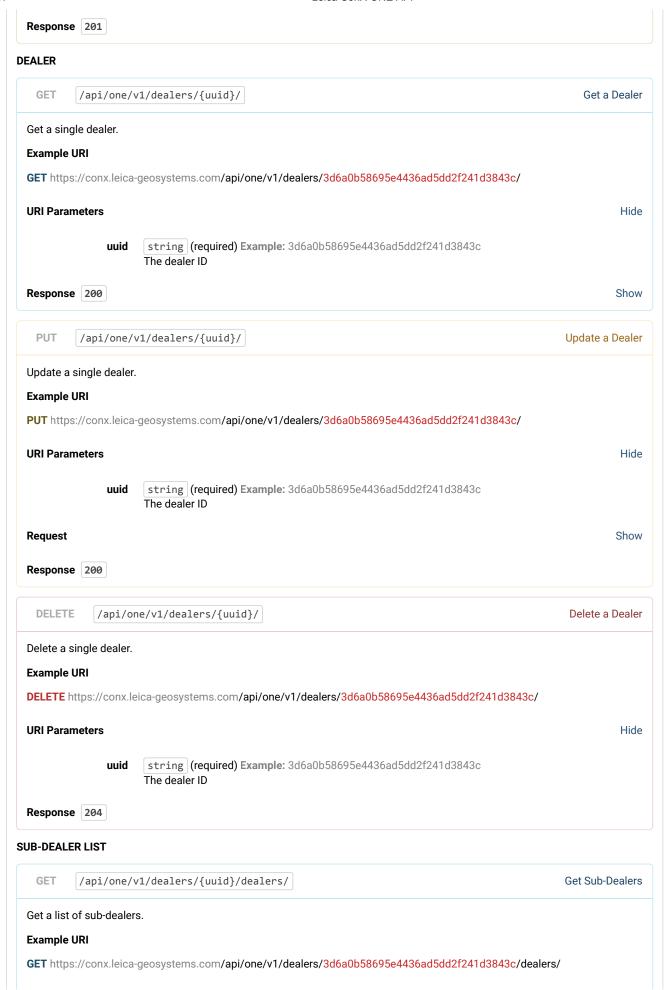
POST

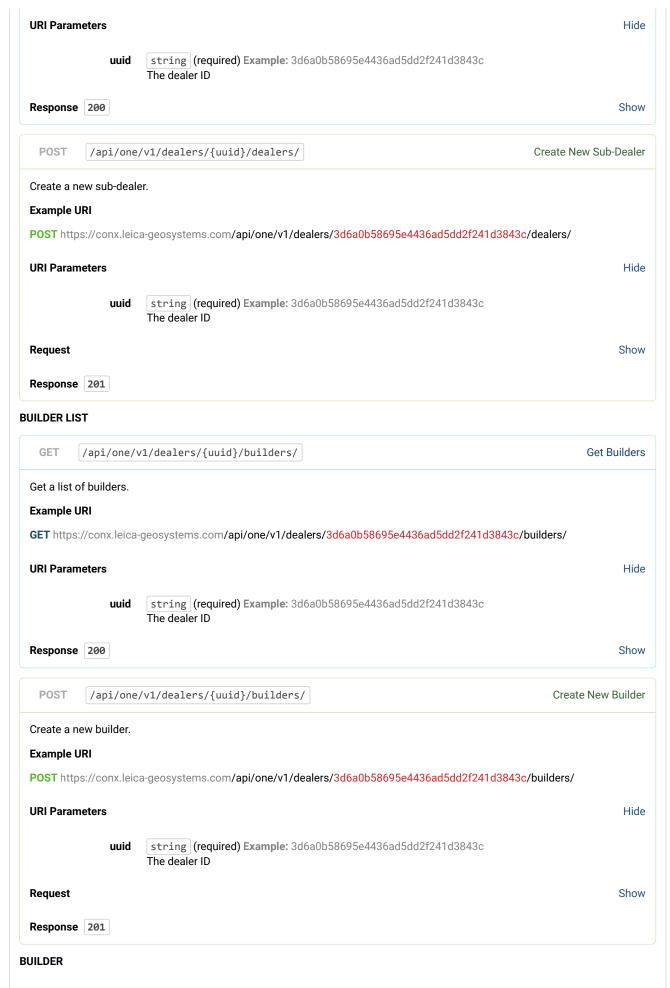
POST https://conx.leica-geosystems.com/api/one/v1/master/dealers/

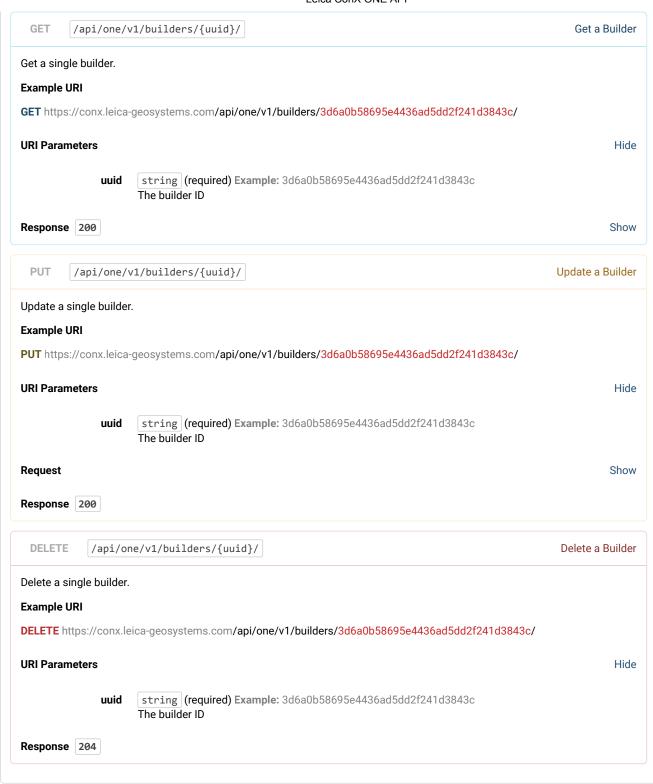
/api/one/v1/master/dealers/

Request Show

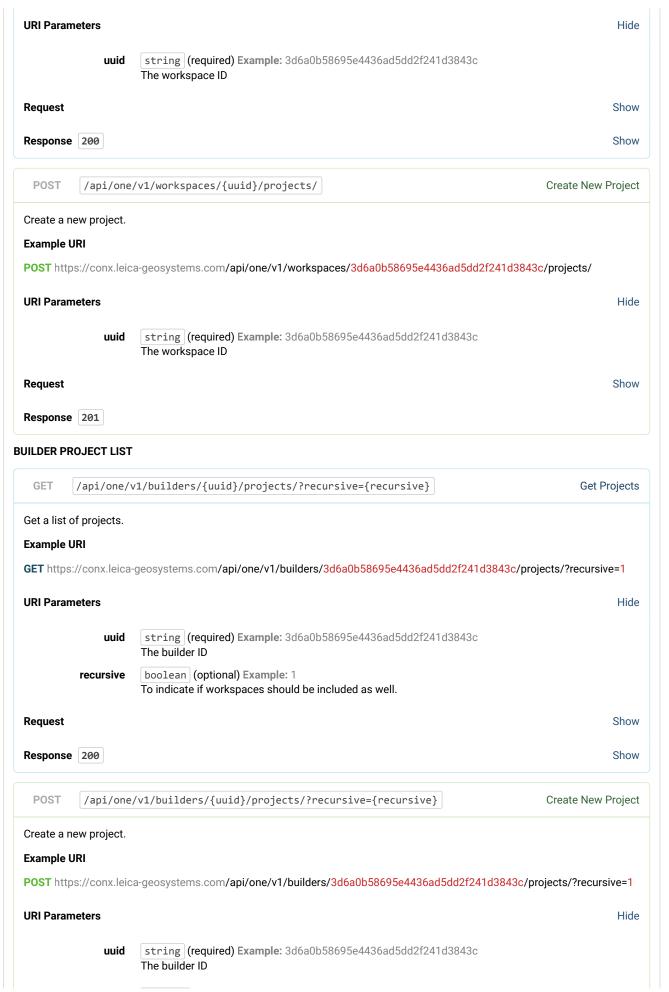
Create New Top-Dealer







WORKSPACE PROJECT LIST GET /api/one/v1/workspaces/{uuid}/projects/ Get a list of projects. Example URI GET https://conx.leica-geosystems.com/api/one/v1/workspaces/3d6a0b58695e4436ad5dd2f241d3843c/projects/



```
recursive boolean (optional) Example: 1
To indicate if workspaces should be included as well.

Request

Response 201
```

Unit

Additional unit metadata available include:

- status last seen, last position and reference model
- · equipment available equipment on unit
- include_discarded include discarded units and equipments
- · with_files include units that have uploaded a file to managed storage
- features waffle flags
- vm_neighbor_data visual machine neighbour data

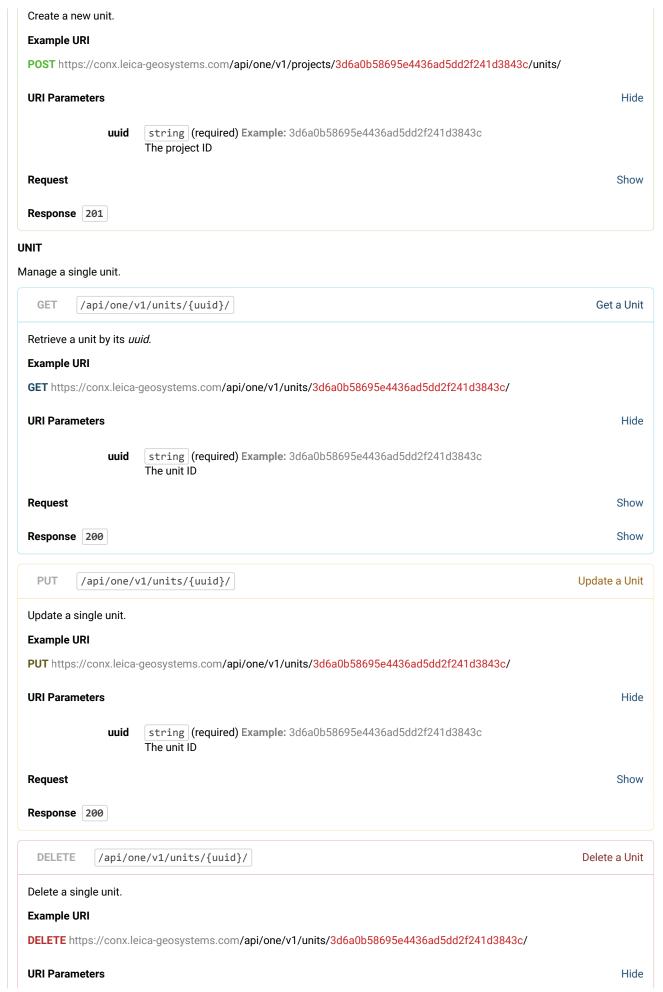
Access the additional metadata by appending ?meta=status, equipment etc to the URL.

Example of metadata:

```
"status": {
    "last_seen": "2015-10-14T10:06:32Z",
    "lat": 12.345,
        "lon": 6.789,
        "altitude": 3.234
    },
    "reference_model": "example.trm",
    "coordinate_system": "example.lok"
},
    "equipment": [
        {
                  "uuid": "3d6a0b58695e4436ad5dd2f241d3843c",
                  "type": "ICON-3D"
        }
        ]
}
```

UNIT LIST

```
/api/one/v1/projects/{uuid}/units/
 GET
                                                                                                        Get Units
Get a list of units
Example URI
GET https://conx.leica-geosystems.com/api/one/v1/projects/3d6a0b58695e4436ad5dd2f241d3843c/units/
URI Parameters
                                                                                                            Hide
                    string (required) Example: 3d6a0b58695e4436ad5dd2f241d3843c
              uuid
                     The project ID
Request
                                                                                                           Show
Response 200
                                                                                                           Show
                                                                                                  Create New Unit
 POST
          /api/one/v1/projects/{uuid}/units/
```



uuid string (required) Example: 3d6a0b58695e4436ad5dd2f241d3843c
The unit ID

Response 204

Equipment

If an equipment in a project is paired once then it can select another project without having to re-pair again. This project selection by the equipment will happen seamlessly and it will start to report sensor data and/or upload/download files from the newly selected project. The following are the steps the equipment has to perform.

PROJECT LIST

Get project list for the equipment.



PROJECT SELECT

Select project to seamlessly connect to.

Please note that after the equipment selects a new project, its uuid will no longer be the same. It will have a new uuid which can be obtained from the "/api/one/v1/me/ endpoint". See Authentication section above for details. Therefore, for subsequent selections of new projects, the equipment uuid have to be re-fetched. However, the equipment token remains the same.

POST /api/one/v1/equipment/{uuid}/select_project/{project_uuid}/ Select project **Example URI** POST https://conx.leica-geosystems.com/api/one/v1/equipment/3d6a0b58695e4436ad5dd2f241d3843c/select_project/1342 f45537e84e28a3e301244ed47523/ **URI Parameters** Hide uuid string (required) Example: 3d6a0b58695e4436ad5dd2f241d3843c The equipment uuid string (required) Example: 1342f45537e84e28a3e301244ed47523 project_uuid The project uuid Request Show Response 200 Show

DeviceState

DEVICESTATE CREATE

A DeviceState entry is a snapshot of the current state of a device.

The DeviceState POST request payload should two keys, a timestamp (observed_at) of when the state was observed on the device, by the device, and a list of currently active files on the device (active_files). What should be considered an active/inactive file is [insert definition here]. The key active files in the payload must be a JSON array, empty or not.

Source files

Each object in active_files can optionally contain a source_file key/value that can be used to describe a chain of file conversions conducted on the device side. In order for the Leica ConX application to keep track of state, these conversions should be communicated through these endpoints when applicable. Note the nested structure in the below example that ends with "source_file": null to indicate the actual source file.

Primary files

Each active_files entry also contains a primary key (of type boolean) that should be used to indicate whether or not a particular file should be considered the "main" file in a collection of used files. This piece of data will be used primarily for frontend display purposes. If this key is not supplied then its value will default to false. A DeviceState entry with a collection of active_files can contain 1 single primary file - attempting to POST a payload containing multiple primary files will return an error response.

MD5 checksum

The md5_checksum key should be an MD5 hash representation of the contents of a given file.

POST /api/one/v1/device-state/

Create a new DeviceState entry.

Example URI

POST https://conx.leica-geosystems.com/api/one/v1/device-state/

Request

Response 201

Data files

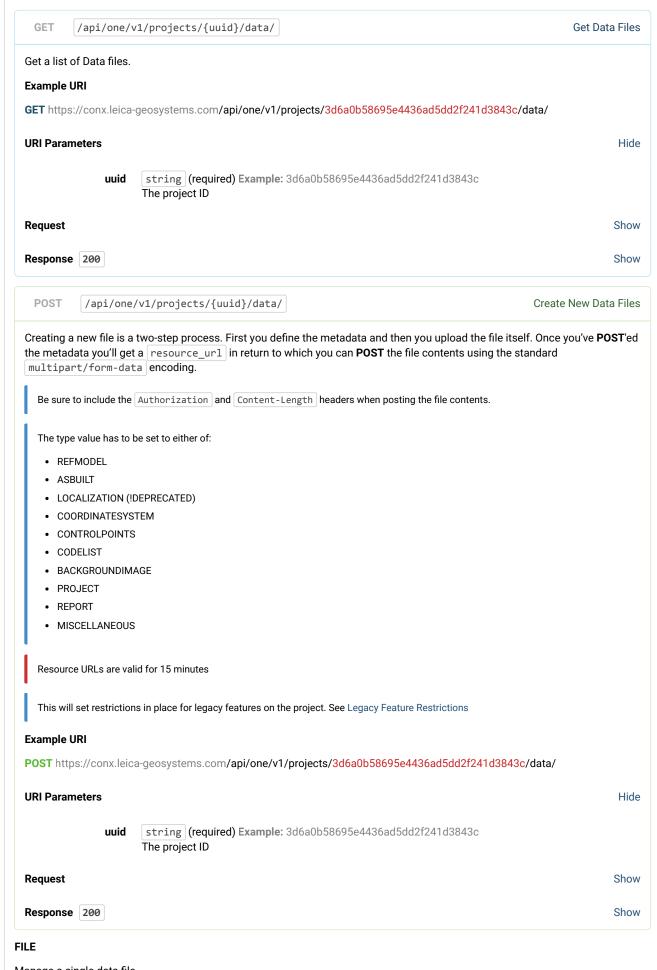
The following data file types are supported:

- · Reference Model
- Asbuilt
- · Localization (Deprecated)
- · Coordinate System (New type deprecating Localization)
- Control Points
- · Code List
- · Background Image
- Project
- Report
- Miscellaneous

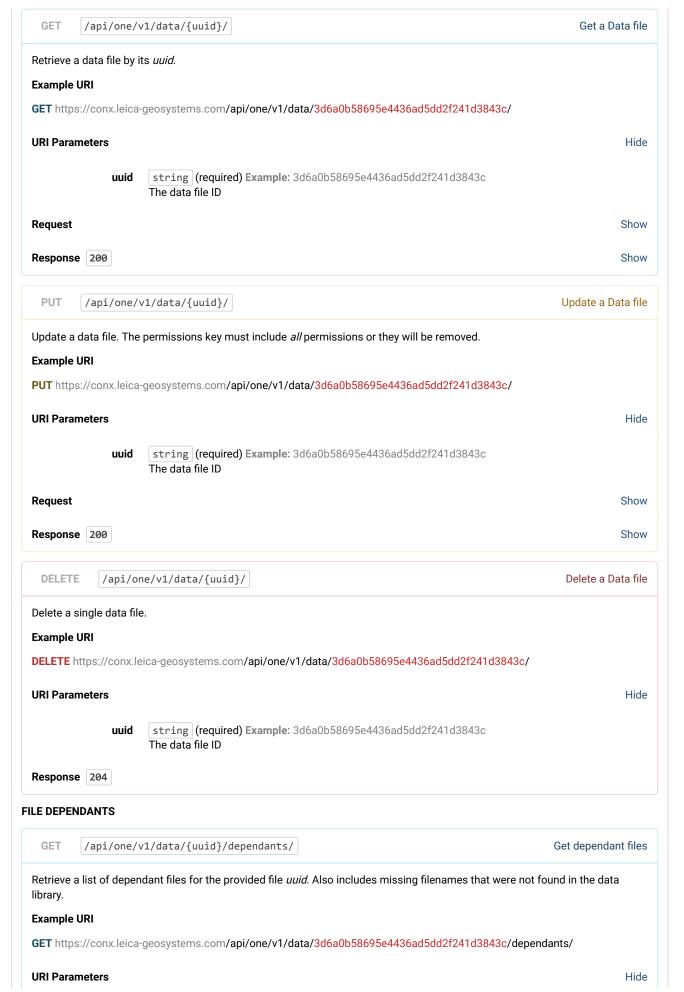
The permissions attribute of a data file object is a list containing uuids of equipment that should have access to the specific data file

DATA FILE LIST

Add url parameter type to list only a certain data file type. The value can be REFMODEL, ASBUILT, LOCALIZATION(!DEPRECATED), COORDINATESYSTEM, CONTROLPOINTS, CODELIST, BACKGROUNDIMAGE, PROJECT, REPORT or MISCELLANEOUS To get all reference models, use: /api/one/v1/projects/{uuid}/data/?type=REFMODEL



Manage a single data file.



uuid string (required) Example: 3d6a0b58695e4436ad5dd2f241d3843c The data file ID to retrieve dependants for Request Show Response 200 Show

FILE EXPORT / CONVERSION



Design

The Design endpoints are deprecated. Please use Data File instead.

The following object classes are being considered:

- · Reference Designs
- Overlays
- · Center line
- · Control points
- · Avoidance zones

The permissions attribute of a design object is a list containing uuids of equipment that should have access to the specific design file

Optional linking to designs available in other GeoJSON objects via the link attribute:

- id ID of the design
- · rel relationship between this survey point and the linked data
- type content type of the linked data
- name suggested persistent storage name for the linked data

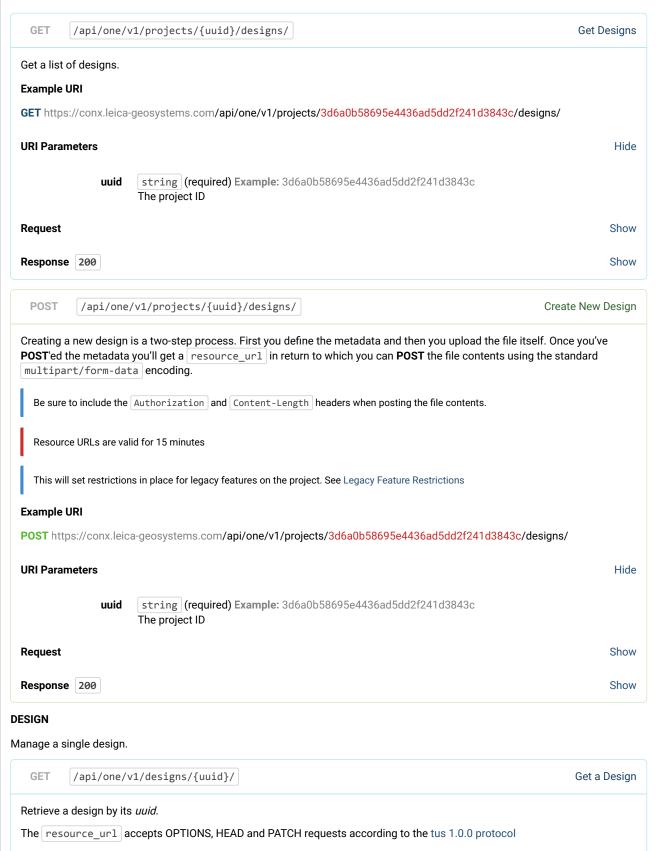
Example of link:

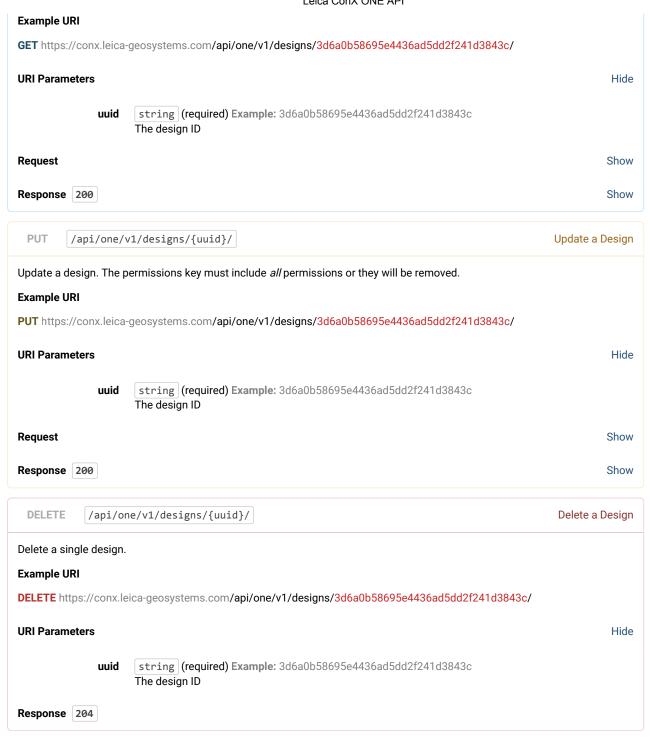
```
"link": {
    "rel": "original",
    "id": "fe3e58c529c74150a84e1f508bd02271",
    "name": "p37.dxf",
    "type" : "application/dxf"
}
```

The state field of the Design object (obtained in the Response JSON) has the following values:

- 0 Waiting for the file content to be uploaded
- · 1 The file is uploaded successfully
- 2 The file is deleted

DESIGN LIST





Survey Point

A survey point is measured by a equipment on the field. The object schema is:

- equipment['uuid'] ID of equipment that measured the point
- unit['uuid'] ID of unit that measured the point
- · date time at which the point was measured
- code selected point code when the point was measured
- tag identifier for the measure point (need not be unique)
- dh delta on the heigth-axis between the measured point and model
- n measure at the northing-axis
- e measure at the easting-axis
- h measure at the height-axis
- model the reference model used (if any)

- · cq GPS quality value
- current_file if the point is in the current file

We have no idea what coordinate system to use at the moment. Should we convert everything to WSG84, or should we maintain the raw coordinate values?

At this time, we know that iCON 3D measures survey points using one of two methods and uploads the result via a file using rsync - the file format depend on the method used:

- stored-point geo (xyz + some selected name)
- autologging csv (xyz + metadata)

For autologging, the metadata include:

- part of the blade/bucket (reference point)
- timestamp
- prefix for the point name (example a machine name)

We are not guaranteed to receive a reference model along with the file - it is assumed that the office user knows the reference model used. Can/should Leica ConX make the same assumption? Or should we require a manual step to "validate" survey points?

EQUIPMENT SURVEY POINT LIST

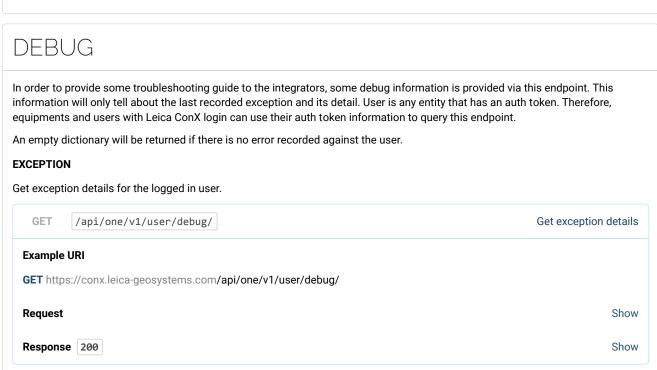
Get list of survey points.

/api/one/v1/equipment/{uuid}/points/{?since} Get Equipment Survey Points Get list of survey points since the given timestamp. **Example URI** GET https://conx.leica-geosystems.com/api/one/v1/equipment/3d6a0b58695e4436ad5dd2f241d3843c/points/?since=2015-0 7-12T14:20:32Z **URI Parameters** Hide string (required) Example: 3d6a0b58695e4436ad5dd2f241d3843c uuid The equipment ID string (required) Example: 2015-07-12T14:20:32Z since Minimum timestamp of points Request Show Response 200 Show

MODEL SURVEY POINT LIST

Get list of survey points.





Generated by aglio on 07 Nov 2017