

# Using the Python REST client

## Python client pyCGA

**pyCGA** is the Python client library for OpenCGA RESTful Web Services, all the web services are accessible through this client, and it offers a quick way to query OpenCGA projects programmatically from custom scripts. In the same way than in [Using RESTful Web Services URL](#) tutorial, we will focus on **those end points more interesting for HGVA users**. In order to make it easy to follow we will use the same examples used in [Using RESTful Web Services URL](#).

## Installing pyCGA

The Python client library is distributed with the rest of the [OpenCGA code](#). The OpenCGA code can be cloned in your machine by executing in your terminal. Checkout the latest code (release-1.1.0 branch). You can easily install pyCGA using *pip* tool:

### How To Install pyCGA

```
git clone https://github.com/opencb/opencga.git
git checkout v1.3.6
cd opencga/opencga-client/src/main/python
[sudo] pip install . [ --upgrade ]
```

## Configuring pyCGA for HGVA

Configuration parameters can be passed as a **JSON** file, **YAML** file or a **Python Dictionary**:

### Configuration File - JSON format

```
{
  "version": "v1",
  "rest": {
    "hosts": [
      "bioinfo.hpc.cam.ac.uk/hgva"
    ]
  }
}
```

### Configuration File - YAML format

```
---
version: v1
rest:
  hosts:
    - bioinfo.hpc.cam.ac.uk/hgva
```

### Table of Contents:

- [Python client pyCGA](#)
  - [Installing pyCGA](#)
  - [Configuring pyCGA for HGVA](#)
- [Examples](#)
  - [Getting information about genomic variants](#)
  - [Getting information about projects](#)
  - [Getting information about studies](#)
  - [Getting information about samples](#)
  - [Getting information about cohorts](#)

### Configuration Dictionary Python

```
configuration = {
    'version': 'v1',
    'rest': {
        'hosts': [
            'bioinfo.hpc.cam.ac.uk/hgva'
        ]
    }
}
```

Load the configuration will be the first step, to use the python client. We will use the **ConfigClient** class, passing the name of the path of the configuration file or the dictionary with the configuration. After that the instance created will be passed to the Client.

### Initialising the client

```
from pyCGA.opencgarestclients import OpenCGAClient

# configuration = '/path/to/configuration_file.json'
# configuration = '/path/to/configuration_file.yaml'
configuration = {
    'version': 'v1',
    'rest': {
        'hosts': [
            'bioinfo.hpc.cam.ac.uk/hgva'
        ]
    }
}

# This will skip the login and allow the user query hgva as Anonymous
oc = OpenCGAClient(configuration=configuration, session_id=' ')
oc.session_id = None
oc._create_clients()
```

Once the library is imported and configured, you can proceed to run the examples below.

## Examples

### Getting information about genomic variants

#### Getting information about genomic variants

```
# Get TTN variants from the Genome of the Netherlands study, which is
# framed within the reference_grch37 project ('limit=3' limit the number of
# results to 3)
# If the response status is 200 (OK), the response will be a dictionary
# with the responses, this dictionary is equivalent to the json response
# obtained through the Web Services.
for page in oc.analysis_variant.query(data={'gene': 'TTN',
    'studies': 'reference_grch37:GONL'}, limit=3, pag_size=100):
    for result in page.get():
        print result
```

### Getting information about projects

#### Getting information about genomic variants

```
# Getting all metadata for the reference_grch37 project
result = oc.projects.info('reference_grch37').get('reference_grch37')

# Getting all studies and their metadata for the cancer_grch37 project
result = oc.projects.studies('reference_grch37').get('reference_grch37')
```

## Getting information about studies

#### Getting information about genomic variants

```
# Getting all metadata for all available studies
responses = oc.studies.search(data={})

# Getting summary data for study 1kG_phase3 which is framed within
project reference_grch37
responses = oc.studies.summary('reference_grch37:1kG_phase3').get(
('reference_grch37:1kG_phase3'))

# Getting all metadata for study GONL which is framed within the project
reference_grch37
responses = oc.studies.info('reference_grch37:GONL').get('reference_grch37:
GONL')

# Getting all samples metadata for study 1kG_phase3 which is framed within
project reference_grch37
responses = oc.studies.samples('reference_grch37:1kG_phase3').get(
('reference_grch37:1kG_phase3'))
```

## Getting information about samples

#### Getting information about genomic variants

```
# Get all metadata for sample HG00096 of the 1kG_phase3 study which is
framed within the reference_grch37 project
responses = oc.samples.info('HG00096', study='reference_grch37:1kG_phase3').
get('HG00096')
```

## Getting information about cohorts

#### Getting information about genomic variants

```
# Get all samples metadata for cohort GBR from study 1kG_phase3 which is
framed within project reference_grch37
responses = oc.cohorts.samples('GBR', study='reference_grch37:1kG_phase3').
get('GBR')
```