Gaapi Documentation

Reck <rakesh.gunduka@gmail.com>

Aug 04, 2021

Contents

A light weight python wrapper for Google's Analytics Reporting API v4 written upon Google API Python Client.

CHAPTER 1

Features provided by Gaapi

- Analytics focussed library to handle reporting api.
- Pythonic style usage.
- Enables you to cache your response data.
- Enables you to get simplified response data. (TO-DO)

CHAPTER 2

Installation

To install, simply use pip or easy_install:

```
$ pip install --upgrade gaapi
```

or:

```
$ easy_install --upgrade gaapi
```

CHAPTER 3

Acquire Google credentials

1. To create a Service Account Credentials, follow the below link

https://support.google.com/a/answer/7378726?hl=en

2. Get view ID in Google Analytics, follow the below link

https://keyword-hero.com/documentation/finding-your-view-id-in-google-analytics

CHAPTER 4

Get Started

4.1 Instanstiate GA Client

```
from gaapi import Client

GA_SERVICE_ACCOUNT_CREDENTIALS = {
   "type": "service_account",
   "project_id": "analytics-xyz",
   "private_key_id": "private_key_id",
   "private_key": "-----BEGIN PRIVATE KEY-----ASADASDONWQENLKQWEIL\nASDASDOILWQE",
   "client_email": "username@analytics-xyz.iam.gserviceaccount.com",
   "client_id": "103486406559549721528",
   "auth_uri": "https://accounts.google.com/o/oauth2/auth",
   "token_uri": "https://accounts.google.com/o/oauth2/token",
   "auth_provider_x509_cert_url": "https://www.googleapis.com/oauth2/v1/certs",
   "client_x509_cert_url": "https://www.googleapis.com/robot/v1/metadata/x509/
↪username@analytics-xyz.iam.gserviceaccount.com"
}
GA_VIEW_ID = '12345567890'


ga = Client(
        credentials=GA_SERVICE_ACCOUNT_CREDENTIALS,
        view_id=GA_VIEW_ID
     )
```

4.2 Generate Query

Gaapi format:

```
query = ga.query.date_ranges(
        start_date='2018-07-03', end_date='today'
    ).metrics(
        expression='ga:users'
    ).dimensions(
        name='ga:pagePath'
    ).dimensions(
        name='ga:eventLabel'
    ).dimension_filter_clauses(
        filters=[{
            "dimension_name": "ga:pagePath",
            "operator": "REGEXP",
            "expressions": ".*something.*"
        }],
    )

# Clone query (Only for Gaapi format)
# Add `clean=True` to reset a value in the query (say date_ranges)
cloned_query = ga.query.clone(query2).date_ranges(
         start_date='2018-08-01', end_date='today', clean=True
    )

# Read query
print(query.json())
```

Python Dictionary format:

```
query = {
    'date_ranges': {
        'start_date': '2018-07-03',
        'end_date': 'today'
    },
    'metrics': [
        {'expression': 'ga:users'}
    ],
    'dimensions': [
        {'name': 'ga:pagePath'}
    ],
    'dimension_filter_clauses': [{
      "filters": [{
          "dimension_name": "ga:pagePath",
          "operator": "REGEXP",
          "expressions": "\/.*something\/.*"
      }]
    }]
}
```

Google's original query format:

```
query = {
    'dateRanges': {
        'startDate': '2018-07-03',
        'endDate': 'today'
    },
    'metrics': [
        {'expression': 'ga:users'}
    ],
```

(continues on next page)

(continued from previous page)

```
    'dimensions': [
        {'name': 'ga:pagePath'}
    ],
    'dimensionFilterClauses': [{
      "filters": [{
          "dimensionName": "ga:pagePath",
          "operator": "REGEXP",
          "expressions": "\/.*something\/.*"
      }]
    }]
}
```

4.3 Request Data

```
# Using gaapi query or python dictionary or json
response = ga.batch_get(query)

# Cache for 600ms
response = ga.batch_get(query, cache_ttl=600)
```

CHAPTER 5

Third Party Libraries and Dependencies

The following external libraries will be installed when you install gaapi . . .

- google-api-python-client
- google-auth
- walrus

CHAPTER 6

To-Dos

- Response Object Manipulation. (This update will enable to you to generate response in Google Raw Response, Simplified Response, CSV, Panda Dataframe).
- Test cases.

CHAPTER 7

Contribute

1. Look for an open issue or create new issue to get a dialog going about the new feature or bug that you've discovered.
2. Fork the repository on Github to start making your changes to the master branch (or branch off of it).
3. Write a test which shows that the bug was fixed or that the feature works as expected.
4. Make a pull request.