
GCM Client Documentation

Release 0.1 beta

Sardar Yumatov

Sep 27, 2017

Contents

1	Requirements	3
2	Alternatives	5
3	Support	7
3.1	Getting Started	7
3.2	gcmclient Package	8
4	Indices and tables	13
	Python Module Index	15

Python client for Google Cloud Messaging (GCM).

Check out the client with similar interface for [Apple Push Notification service](#).

CHAPTER 1

Requirements

- [requests](#) - HTTP request, handles proxies etc.
- [omnijson](#) if you use Python 2.5 or older.

The only alternative library known at the time of writing was `python-gcm`. This library differs in the following design decisions:

- *Predictable execution time.* Do not automatically retry request on failure. According to Google's recommendations, each retry has to wait exponential back-off delay. We use Celery back-end, where the best way to retry after some delay will be scheduling the task with `countdown=delay`. Sleeping while in Celery worker hurts your concurrency.
- *Do not forget results if you need to retry.* This sounds obvious, but `python-gcm` drops important results, such as canonical ID mapping if request needs to be (partially) retried.
- *Clean pythonic API.* No need to borrow all Java like exceptions etc.
- *Do not hard-code validation, let GCM fail.* This decision makes library a little bit more future proof.

GCM client was created by [Sardar Yumatov](#), contact me if you find any bugs or need help. Contact [Getlogic](#) if you need a full-featured push notification service for all popular platforms. You can view outstanding issues on the [GCM Bitbucket](#) page.

Contents:

Getting Started

You need Google API key in order to consume Google's services. You can obtain such key from the [developers console](#). Open *Services* section and switch on *Google Cloud Messaging for Android*. Then open *API Access* section and create *Key for server apps* if you haven't any. The *API key* string is what you need. Ensure IP filter is disabled or your server IP is listed.

Consult [Google Cloud Messaging for Android](#) for all options that you might pass with each message. There you will also find all error codes, such as `MismatchSenderId`, that can be returned by GCM.

Usage

Usage is straightforward:

```
from gcmclient import *

# Pass 'proxies' keyword argument, as described in 'requests' library if you
# use proxies. Check other options too.
gcm = GCM(API_KEY)

# Construct (key => scalar) payload. do not use nested structures.
data = {'str': 'string', 'int': 10}

# Unicast or multicast message, read GCM manual about extra options.
# It is probably a good idea to always use JSONMessage, even if you send
# a notification to just 1 registration ID.
```

```
unicast = PlainTextMessage("registration_id", data, dry_run=True)
multicast = JSONMessage(["registration_id_1", "registration_id_2"], data, collapse_
↳key='my.key', dry_run=True)

try:
    # attempt send
    res_unicast = gcm.send(unicast)
    res_multicast = gcm.send(multicast)

    for res in [res_unicast, res_multicast]:
        # nothing to do on success
        for reg_id, msg_id in res.success.items():
            print "Successfully sent %s as %s" % (reg_id, msg_id)

        # update your registration ID's
        for reg_id, new_reg_id in res.canonical.items():
            print "Replacing %s with %s in database" % (reg_id, new_reg_id)

        # probably app was uninstalled
        for reg_id in res.not_registered:
            print "Removing %s from database" % reg_id

        # unrecoverably failed, these ID's will not be retried
        # consult GCM manual for all error codes
        for reg_id, err_code in res.failed.items():
            print "Removing %s because %s" % (reg_id, err_code)

        # if some registration ID's have recoverably failed
        if res.needs_retry():
            # construct new message with only failed regids
            retry_msg = res.retry()
            # you have to wait before attempting again. delay()
            # will tell you how long to wait depending on your
            # current retry counter, starting from 0.
            print "Wait or schedule task after %s seconds" % res.delay(retry)
            # retry += 1 and send retry_msg again

except GCMAuthenticationError:
    # stop and fix your settings
    print "Your Google API key is rejected"
except ValueError, e:
    # probably your extra options, such as time_to_live,
    # are invalid. Read error message for more info.
    print "Invalid message/option or invalid GCM response"
    print e.args[0]
except Exception:
    # your network is down or maybe proxy settings
    # are broken. when problem is resolved, you can
    # retry the whole message.
    print "Something wrong with requests library"
```

gcmclient Package

Google Cloud Messaging client built on top of requests library.

gcmclient Package

`gcmclient.gcm.GCM_URL = 'https://android.googleapis.com/gcm/send'`
Default URL to GCM service.

class `gcmclient.gcm.GCM` (*api_key*, *url*='https://android.googleapis.com/gcm/send', *backoff*=1000, ****options**)
Create new connection.

Arguments

- *api_key* (str): Google API key.
- *url* (str): GCM server URL.
- *backoff* (int): initial backoff in milliseconds.
- *options* (kwargs): options for `requests` such as proxies.

send (*message*)

Send message.

The message may contain various options, such as `time_to_live`. Your request might be rejected, because some of your options might be invalid. In this case a `ValueError` with explanation will be raised.

Arguments *message* (`Message`): plain text or JSON message.

Returns `Result` interpreting the results.

Raises

- `requests.exceptions.RequestException` on any network problem.
- `ValueError` if your GCM request or response is rejected.
- `GCMAuthenticationError` your API key is invalid.

class `gcmclient.gcm.JSONMessage` (*registration_ids*, *data*=None, ****options**)
Multicast message, uses JSON format.

Arguments

- *registration_ids* (list): registration ID's of target devices.
- *data* (dict): key-value pairs of message payload.
- *options* (kwargs): GCM options, see `Message` for more info.

__getstate__ ()

Returns dict with `__init__` arguments.

If you use `pickle`, then simply `pickle/unpickle` the message object. If you use something else, like JSON, then:

```
# obtain state dict from message
state = message.__getstate__()
# send/store the state
# recover state and restore message. you have to pick the right class
message_copy = JSONMessage(**state)
```

Returns *kwargs* for `JSONMessage` constructor.

registration_ids

Target registration ID's.

class `gcmclient.gcm.PlainTextMessage` (*registration_id*, *data=None*, ***options*)
Unicast message, uses plain text format. All values in the data payload must be URL encodable scalars.

Arguments

- *registration_id* (str): registration ID of target device.
- *data* (dict): key-value pairs of message payload.
- *options* (kwargs): GCM options, see *Message* for more info.

`__getstate__()`

Returns dict with `__init__` arguments.

If you use `pickle`, then simply `pickle/unpickle` the message object. If you use something else, like JSON, then:

```
# obtain state dict from message
state = message.__getstate__()
# send/store the state
# recover state and restore message. you have to pick the right class
message_copy = PlainTextMessage(**state)
```

Returns *kwargs* for *PlainTextMessage* constructor.

registration_id

Target registration ID.

class `gcmclient.gcm.Message` (*data=None*, *options=None*)
Abstract message.

Arguments

- *data* (dict): key-value pairs, payload of this message.
- *options* (dict): GCM options.

Refer to [GCM](#) for more explanation on available options.

Options

- *collapse_key* (str): collapse key/bucket.
- *time_to_live* (int): message TTL in seconds.
- *delay_while_idle* (bool): hold message if device is off-line.
- *restricted_package_name* (str): declare package name.
- *dry_run* (bool): pretend sending message to devices.

class `gcmclient.gcm.Result` (*message*, *response*, *backoff*)
Result of send operation.

You should check `canonical()` for any registration ID's that should be updated. If the whole message or some registration ID's have recoverably failed, then `retry()` will provide you with new message. You have to wait `delay()` seconds before attempting a new request.

backoff (*retry=0*)

Computes exponential backoff for given retry number.

canonical

New registration ID's as mapping `{registration_id: canonical_id}`.

You have to update registration ID's of your subscribers by replacing them with corresponding canonical ID. Read more [here](#).

delay (*retry=0*)

Time to wait in seconds before attempting a retry as a float number.

This method will return value of Retry-After header if it is provided by GCM. Otherwise, it will return (backoff * 2^retry) with some random shift. Google may black list your server if you do not honor Retry-After hint and do not use exponential backoff.

failed

Unrecoverably failed registration ID's as mapping {registration_id: error code}.

This method lists devices, that have failed with something else than:

- Unavailable – look for *retry()* instead.
- NotRegistered – look for *not_registered* instead.

Read more about possible [error codes](#).

needs_retry ()

True if *retry()* will return message.

not_registered

List all registration ID's that GCM reports as NotRegistered. You should remove them from your database.

retry ()

Construct new message that will unicast/multicast to remaining recoverably failed registration ID's. Method returns None if there is nothing to retry. Do not forget to wait for *delay()* seconds before new attempt.

success

Successfully processed registration ID's as mapping {registration_id: message_id}.

class gcmclient.gcm.GCMAuthenticationError

Raised if your Google API key is rejected.

CHAPTER 4

Indices and tables

- `genindex`
- `modindex`
- `search`

g

`gcmclient.gcm`, 9

Symbols

`__getstate__()` (gcmclient.gcm.JSONMessage method), 9
`__getstate__()` (gcmclient.gcm.PlainTextMessage method), 10

B

`backoff()` (gcmclient.gcm.Result method), 10

C

`canonical` (gcmclient.gcm.Result attribute), 10

D

`delay()` (gcmclient.gcm.Result method), 11

F

`failed` (gcmclient.gcm.Result attribute), 11

G

GCM (class in gcmclient.gcm), 9
GCM_URL (in module gcmclient.gcm), 9
GCMAuthenticationError (class in gcmclient.gcm), 11
gcmclient.gcm (module), 9

J

JSONMessage (class in gcmclient.gcm), 9

M

Message (class in gcmclient.gcm), 10

N

`needs_retry()` (gcmclient.gcm.Result method), 11
`not_registered` (gcmclient.gcm.Result attribute), 11

P

PlainTextMessage (class in gcmclient.gcm), 9

R

`registration_id` (gcmclient.gcm.PlainTextMessage attribute), 10

`registration_ids` (gcmclient.gcm.JSONMessage attribute), 9

Result (class in gcmclient.gcm), 10

`retry()` (gcmclient.gcm.Result method), 11

S

`send()` (gcmclient.gcm.GCM method), 9

`success` (gcmclient.gcm.Result attribute), 11