
gCloud Documentation

Release 0.10.0

Google Cloud Platform

September 08, 2016

1	Base Client	1
2	Credentials Helpers	5
3	Base Connections	9
4	Exceptions	13
5	Environment Variables	17
6	Authentication	19
6.1	Overview	19
6.2	Client-Provided Authentication	19
6.3	Explicit Credentials	20
6.4	Troubleshooting	21
6.5	Advanced Customization	22
7	Datastore Client	25
7.1	Connection	27
8	Entities	31
9	Keys	33
10	Queries	37
11	Transactions	41
12	Batches	45
13	Helpers	49
14	Storage Client	51
14.1	Connection	53
15	Blobs / Objects	55
16	Buckets	63
17	ACL	71

18 Batches	77
19 Using the API	79
19.1 Authentication / Configuration	79
19.2 Manage topics for a project	79
19.3 Publish messages to a topic	80
19.4 Manage subscriptions to topics	80
19.5 Pull messages from a subscription	82
20 Pub/Sub Client	83
20.1 Connection	84
21 Topics	87
22 Subscriptions	91
23 Message	95
24 Using the API	97
24.1 Authentication / Configuration	97
24.2 Projects	97
24.3 Datasets	98
24.4 Tables	99
24.5 Jobs	100
25 BigQuery Client	107
25.1 Connection	109
26 Datasets	111
27 Jobs	115
28 Tables	121
29 Query	127
30 Resource Manager Overview	131
30.1 Authentication	132
31 Client	133
31.1 Connection	134
32 Projects	137
33 Using the API	141
33.1 Client	141
33.2 Projects	141
33.3 Project Quotas	141
33.4 Managed Zones	142
33.5 Resource Record Sets	142
33.6 Change requests	143
34 DNS Client	145
34.1 Connection	146
35 Managed Zones	147

36 Resource Record Sets	151
37 Change Sets	153
38 Using the API	155
38.1 Overview	155
38.2 Indexes	155
38.3 Documents	156
38.4 Fields	157
38.5 Searching	157
39 Search Client	159
39.1 Connection	160
40 Indexes	161
41 Documents	165
42 Getting started	169
42.1 Cloud Datastore	169
42.2 Cloud Storage	169
Python Module Index	171

Base Client

Base classes for client used to interact with Google Cloud APIs.

class `gcloud.client.Client` (*credentials=None, http=None*)

Bases: `gcloud.client._ClientFactoryMixin`

Client to bundle configuration needed for API requests.

Assumes that the associated `_connection_class` only accepts `http` and `credentials` in its constructor.

Parameters

- **credentials** (`oauth2client.client.OAuth2Credentials` or `NoneType`) – The OAuth2 Credentials to use for the connection owned by this client. If not passed (and if no `http` object is passed), falls back to the default inferred from the environment.
- **http** (`httplib2.Http` or class that defines `request()`) – An optional HTTP object to make requests. If not passed, an `http` object is created that is bound to the `credentials` for the current object.

from_service_account_json (*json_credentials_path, *args, **kwargs*)

Factory to retrieve JSON credentials while creating client.

Parameters

- **json_credentials_path** (*string*) – The path to a private key file (this file was given to you when you created the service account). This file must contain a JSON object with a private key and other credentials information (downloaded from the Google APIs console).
- **args** (*tuple*) – Remaining positional arguments to pass to constructor.
- **kwargs** (*dict*) – Remaining keyword arguments to pass to constructor.

Return type `gcloud.pubsub.client.Client`

Returns The client created with the retrieved JSON credentials.

Raises `TypeError` if there is a conflict with the `kwargs` and the credentials created by the factory.

from_service_account_p12 (*client_email, private_key_path, *args, **kwargs*)

Factory to retrieve P12 credentials while creating client.

Note: Unless you have an explicit reason to use a PKCS12 key for your service account, we recommend using a JSON key.

Parameters

- **client_email** (*string*) – The e-mail attached to the service account.
- **private_key_path** (*string*) – The path to a private key file (this file was given to you when you created the service account). This file must be in P12 format.
- **args** (*tuple*) – Remaining positional arguments to pass to constructor.
- **kwargs** (*dict*) – Remaining keyword arguments to pass to constructor.

Return type `gcloud.client.Client`

Returns The client created with the retrieved P12 credentials.

Raises `TypeError` if there is a conflict with the kwargs and the credentials created by the factory.

class `gcloud.client.JSONClient` (*project=None, credentials=None, http=None*)
Bases: `gcloud.client.Client`, `gcloud.client._ClientProjectMixin`

Client to for Google JSON-based API.

Assumes such APIs use the `project` and the client needs to store this value.

Parameters

- **project** (*string*) – the project which the client acts on behalf of. If not passed falls back to the default inferred from the environment.
- **credentials** (`oauth2client.client.OAuth2Credentials` or `NoneType`) – The OAuth2 Credentials to use for the connection owned by this client. If not passed (and if no `http` object is passed), falls back to the default inferred from the environment.
- **http** (`httplib2.Http` or class that defines `request()`.) – An optional HTTP object to make requests. If not passed, an `http` object is created that is bound to the `credentials` for the current object.

Raises `ValueError` if the project is neither passed in nor set in the environment.

from_service_account_json (*json_credentials_path, *args, **kwargs*)

Factory to retrieve JSON credentials while creating client.

Parameters

- **json_credentials_path** (*string*) – The path to a private key file (this file was given to you when you created the service account). This file must contain a JSON object with a private key and other credentials information (downloaded from the Google APIs console).
- **args** (*tuple*) – Remaining positional arguments to pass to constructor.
- **kwargs** (*dict*) – Remaining keyword arguments to pass to constructor.

Return type `gcloud.pubsub.client.Client`

Returns The client created with the retrieved JSON credentials.

Raises `TypeError` if there is a conflict with the kwargs and the credentials created by the factory.

from_service_account_p12 (*client_email*, *private_key_path*, *args, **kwargs)
Factory to retrieve P12 credentials while creating client.

Note: Unless you have an explicit reason to use a PKCS12 key for your service account, we recommend using a JSON key.

Parameters

- **client_email** (*string*) – The e-mail attached to the service account.
- **private_key_path** (*string*) – The path to a private key file (this file was given to you when you created the service account). This file must be in P12 format.
- **args** (*tuple*) – Remaining positional arguments to pass to constructor.
- **kwargs** (*dict*) – Remaining keyword arguments to pass to constructor.

Return type *gcloud.client.Client*

Returns The client created with the retrieved P12 credentials.

Raises `TypeError` if there is a conflict with the kwargs and the credentials created by the factory.

Credentials Helpers

A simple wrapper around the OAuth2 credentials library.

```
gcloud.credentials.generate_signed_url(credentials, resource, expiration,
                                       api_access_endpoint='', method='GET',
                                       content_md5=None, content_type=None, re-
                                       sponse_type=None, response_disposition=None,
                                       generation=None)
```

Generate signed URL to provide query-string auth'n to a resource.

Note: If you are on Google Compute Engine, you can't generate a signed URL. Follow [Issue 922](#) for updates on this. If you'd like to be able to generate a signed URL from GCE, you can use a standard service account from a JSON file rather than a GCE service account.

See headers [reference](#) for more details on optional arguments.

Parameters

- **credentials** (`oauth2client.appengine.AppAssertionCredentials`) – Credentials object with an associated private key to sign text.
- **resource** (`string`) – A pointer to a specific resource (typically, `/bucket-name/path/to/blob.txt`).
- **expiration** (`int`, `long`, `datetime.datetime`, `datetime.timedelta`) – When the signed URL should expire.
- **api_access_endpoint** (`str`) – Optional URI base. Defaults to empty string.
- **method** (`str`) – The HTTP verb that will be used when requesting the URL. Defaults to `'GET'`.
- **content_md5** (`str`) – (Optional) The MD5 hash of the object referenced by `resource`.
- **content_type** (`str`) – (Optional) The content type of the object referenced by `resource`.
- **response_type** (`str`) – (Optional) Content type of responses to requests for the signed URL. Used to over-ride the content type of the underlying resource.
- **response_disposition** (`str`) – (Optional) Content disposition of responses to requests for the signed URL.
- **generation** (`str`) – (Optional) A value that indicates which generation of the resource to fetch.

Return type `string`

Returns A signed URL you can use to access the resource until expiration.

`gcloud.credentials.get_credentials()`
Gets credentials implicitly from the current environment.

Note: You should not need to use this function directly. Instead, use a helper method which uses this method under the hood.

Checks environment in order of precedence:

- Google App Engine (production and testing)
- Environment variable `GOOGLE_APPLICATION_CREDENTIALS` pointing to a file with stored credentials information.
- Stored “well known” file associated with `gcloud` command line tool.
- Google Compute Engine production environment.

The file referred to in `GOOGLE_APPLICATION_CREDENTIALS` is expected to contain information about credentials that are ready to use. This means either service account information or user account information with a ready-to-use refresh token:

```
{
  'type': 'authorized_user',
  'client_id': '...',
  'client_secret': '...',
  'refresh_token': '...'
}
```

or

```
{
  'type': 'service_account',
  'client_id': '...',
  'client_email': '...',
  'private_key_id': '...',
  'private_key': '...'
}
```

The second of these is simply a JSON key downloaded from the Google APIs console. The first is a close cousin of the “client secrets” JSON file used by `oauth2client.clientsecrets` but differs in formatting.

Return type `oauth2client.client.GoogleCredentials`,
`oauth2client.contrib.appengine.AppAssertionCredentials`,
`oauth2client.contrib.gce.AppAssertionCredentials`,
`oauth2client.service_account.ServiceAccountCredentials`

Returns A new credentials instance corresponding to the implicit environment.

`gcloud.credentials.get_for_service_account_json(json_credentials_path, scope=None)`
Gets the credentials for a service account with JSON key.

Parameters

- **json_credentials_path** (*string*) – The path to a private key file (this file was given to you when you created the service account). This file must contain a JSON object with a private key and other credentials information (downloaded from the Google APIs console).

- **scope** (*string or tuple of string*) – The scope against which to authenticate. (Different services require different scopes, check the documentation for which scope is required for the different levels of access to any particular API.)

Return type `oauth2client.client.GoogleCredentials`,
`oauth2client.service_account.ServiceAccountCredentials`

Returns New service account or Google (for a user JSON key file) credentials object.

`gcloud.credentials.get_for_service_account_p12` (*client_email*, *private_key_path*,
scope=None)

Gets the credentials for a service account with PKCS12 / p12 key.

Note: This method is not used by default, instead `get_credentials()` is used. This method is intended to be used when the environment is known explicitly and detecting the environment implicitly would be superfluous.

Parameters

- **client_email** (*string*) – The e-mail attached to the service account.
- **private_key_path** (*string*) – The path to a private key file (this file was given to you when you created the service account). This file must be in P12 format.
- **scope** (*string or tuple of string*) – The scope against which to authenticate. (Different services require different scopes, check the documentation for which scope is required for the different levels of access to any particular API.)

Return type `oauth2client.service_account.ServiceAccountCredentials`

Returns A new `ServiceAccountCredentials` instance with the needed service account settings.

Base Connections

Shared implementation of connections to API servers.

`gcloud.connection.API_BASE_URL = 'https://www.googleapis.com'`
The base of the API call URL.

class `gcloud.connection.Connection` (*credentials=None, http=None*)
Bases: `object`

A generic connection to Google Cloud Platform.

Subclasses should understand only the basic types in method arguments, however they should be capable of returning advanced types.

If no value is passed in for `http`, a `httplib2.Http` object will be created and authorized with the `credentials`. If not, the `credentials` and `http` need not be related.

Subclasses may seek to use the private key from `credentials` to sign data.

A custom (non-`httplib2`) HTTP object must have a `request` method which accepts the following arguments:

- `uri`
- `method`
- `body`
- `headers`

In addition, `redirections` and `connection_type` may be used.

Without the use of `credentials.authorize(http)`, a custom `http` object will also need to be able to add a bearer token to API requests and handle token refresh on 401 errors.

Parameters

- **`credentials`** (`oauth2client.client.OAuth2Credentials` or `NoneType`) – The OAuth2 Credentials to use for this connection.
- **`http`** (`httplib2.Http` or class that defines `request()`) – An optional HTTP object to make requests.

SCOPE = None

The scopes required for authenticating with a service.

Needs to be set by subclasses.

USER_AGENT = 'gcloud-python/0.10.0'

The user agent for gcloud-python requests.

credentials

Getter for current credentials.

Return type `oauth2client.client.OAuth2Credentials` or `NoneType`

Returns The credentials object associated with this connection.

http

A getter for the HTTP transport used in talking to the API.

Return type `httplib2.Http`

Returns A `Http` object used to transport data.

class `gcloud.connection.JSONConnection` (*credentials=None, http=None*)

Bases: `gcloud.connection.Connection`

A connection to a Google JSON-based API.

These APIs are discovery based. For reference:

<https://developers.google.com/discovery/>

This defines `api_request()` for making a generic JSON API request and API requests are created elsewhere.

The class constants

- `API_BASE_URL`
- `API_VERSION`
- `API_URL_TEMPLATE`

must be updated by subclasses.

API_BASE_URL = None

The base of the API call URL.

API_URL_TEMPLATE = None

A template for the URL of a particular API call.

API_VERSION = None

The version of the API, used in building the API call's URL.

api_request (*method, path, query_params=None, data=None, content_type=None, api_base_url=None, api_version=None, expect_json=True, _target_object=None*)

Make a request over the HTTP transport to the API.

You shouldn't need to use this method, but if you plan to interact with the API using these primitives, this is the correct one to use.

Parameters

- **method** (*string*) – The HTTP method name (ie, GET, POST, etc). Required.
- **path** (*string*) – The path to the resource (ie, `'/b/bucket-name'`). Required.
- **query_params** (*dict*) – A dictionary of keys and values to insert into the query string of the URL. Default is empty dict.
- **data** (*string*) – The data to send as the body of the request. Default is the empty string.
- **content_type** (*string*) – The proper MIME type of the data provided. Default is `None`.
- **api_base_url** (*string*) – The base URL for the API endpoint. Typically you won't have to provide this. Default is the standard API base URL.

- **api_version** (*string*) – The version of the API to call. Typically you shouldn't provide this and instead use the default for the library. Default is the latest API version supported by gcloud-python.
- **expect_json** (*bool*) – If True, this method will try to parse the response as JSON and raise an exception if that cannot be done. Default is True.
- **_target_object** (*object* or `NoneType`) – Protected argument to be used by library callers. This can allow custom behavior, for example, to defer an HTTP request and complete initialization of the object at a later time.

Raises Exception if the response code is not 200 OK.

classmethod **build_api_url** (*path*, *query_params=None*, *api_base_url=None*, *api_version=None*)

Construct an API url given a few components, some optional.

Typically, you shouldn't need to use this method.

Parameters

- **path** (*string*) – The path to the resource (ie, `'/b/bucket-name'`).
- **query_params** (*dict*) – A dictionary of keys and values to insert into the query string of the URL.
- **api_base_url** (*string*) – The base URL for the API endpoint. Typically you won't have to provide this.
- **api_version** (*string*) – The version of the API to call. Typically you shouldn't provide this and instead use the default for the library.

Return type *string*

Returns The URL assembled from the pieces provided.

Exceptions

Custom exceptions for `gcloud` package.

See: https://cloud.google.com/storage/docs/json_api/v1/status-codes

exception `gcloud.exceptions.BadRequest` (*message*, *errors*=())

Bases: `gcloud.exceptions.ClientError`

Exception mapping a '400 Bad Request' response.

code = 400

exception `gcloud.exceptions.ClientError` (*message*, *errors*=())

Bases: `gcloud.exceptions.GCloudError`

Base for 4xx responses

This class is abstract

exception `gcloud.exceptions.Conflict` (*message*, *errors*=())

Bases: `gcloud.exceptions.ClientError`

Exception mapping a '409 Conflict' response.

code = 409

exception `gcloud.exceptions.Forbidden` (*message*, *errors*=())

Bases: `gcloud.exceptions.ClientError`

Exception mapping a '403 Forbidden' response.

code = 403

exception `gcloud.exceptions.GCloudError` (*message*, *errors*=())

Bases: `exceptions.Exception`

Base error class for `gcloud` errors (abstract).

Each subclass represents a single type of HTTP error response.

code = None

HTTP status code. Concrete subclasses *must* define.

See: <http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html>

errors

Detailed error information.

Return type list(dict)

Returns a list of mappings describing each error.

exception `gcloud.exceptions.InternalServerError` (*message*, *errors*=())

Bases: `gcloud.exceptions.ServerError`

Exception mapping a '500 Internal Server Error' response.

code = 500

exception `gcloud.exceptions.LengthRequired` (*message*, *errors*=())

Bases: `gcloud.exceptions.ClientError`

Exception mapping a '411 Length Required' response.

code = 411

exception `gcloud.exceptions.MethodNotAllowed` (*message*, *errors*=())

Bases: `gcloud.exceptions.ClientError`

Exception mapping a '405 Method Not Allowed' response.

code = 405

exception `gcloud.exceptions.MovedPermanently` (*message*, *errors*=())

Bases: `gcloud.exceptions.Redirection`

Exception mapping a '301 Moved Permanently' response.

code = 301

exception `gcloud.exceptions.NotFound` (*message*, *errors*=())

Bases: `gcloud.exceptions.ClientError`

Exception mapping a '404 Not Found' response.

code = 404

exception `gcloud.exceptions.NotImplemented` (*message*, *errors*=())

Bases: `gcloud.exceptions.ServerError`

Exception mapping a '501 Not Implemented' response.

code = 501

exception `gcloud.exceptions.NotModified` (*message*, *errors*=())

Bases: `gcloud.exceptions.Redirection`

Exception mapping a '304 Not Modified' response.

code = 304

exception `gcloud.exceptions.PreconditionFailed` (*message*, *errors*=())

Bases: `gcloud.exceptions.ClientError`

Exception mapping a '412 Precondition Failed' response.

code = 412

exception `gcloud.exceptions.Redirection` (*message*, *errors*=())

Bases: `gcloud.exceptions.GCloudError`

Base for 3xx responses

This class is abstract.

exception `gcloud.exceptions.RequestRangeNotSatisfiable` (*message*, *errors*=())

Bases: `gcloud.exceptions.ClientError`

Exception mapping a '416 Request Range Not Satisfiable' response.

code = 416

exception `gcloud.exceptions.ResumeIncomplete` (*message*, *errors*=())

Bases: `gcloud.exceptions.Redirection`

Exception mapping a '308 Resume Incomplete' response.

code = 308

exception `gcloud.exceptions.ServerError` (*message*, *errors*=())

Bases: `gcloud.exceptions.GCloudError`

Base for 5xx responses: (abstract)

exception `gcloud.exceptions.ServiceUnavailable` (*message*, *errors*=())

Bases: `gcloud.exceptions.ServerError`

Exception mapping a '503 Service Unavailable' response.

code = 503

exception `gcloud.exceptions.TemporaryRedirect` (*message*, *errors*=())

Bases: `gcloud.exceptions.Redirection`

Exception mapping a '307 Temporary Redirect' response.

code = 307

exception `gcloud.exceptions.TooManyRequests` (*message*, *errors*=())

Bases: `gcloud.exceptions.ClientError`

Exception mapping a '429 Too Many Requests' response.

code = 429

exception `gcloud.exceptions.Unauthorized` (*message*, *errors*=())

Bases: `gcloud.exceptions.ClientError`

Exception mapping a '401 Unauthorized' response.

code = 401

`gcloud.exceptions.make_exception` (*response*, *content*, *error_info*=None, *use_json*=True)

Factory: create exception based on HTTP response code.

Parameters

- **response** (`httplib2.Response` or other HTTP response object) – A response object that defines a status code as the status attribute.
- **content** (*string* or *dictionary*) – The body of the HTTP error response.
- **error_info** (*string*) – Optional string giving extra information about the failed request.
- **use_json** (*bool*) – Flag indicating if content is expected to be JSON.

Return type instance of `GCloudError`, or a concrete subclass.

Returns Exception specific to the error response.

Environment Variables

Comprehensive list of environment variables used in gcloud.

These enable many types of implicit behavior in both production and tests.

`gcloud.environment_vars.CREDENTIALS = 'GOOGLE_APPLICATION_CREDENTIALS'`

Environment variable defining location of Google credentials.

`gcloud.environment_vars.DATASET = 'GLOUD_DATASET_ID'`

Environment variable defining default dataset ID.

`gcloud.environment_vars.GCD_DATASET = 'DATASTORE_DATASET'`

Environment variable defining default dataset ID under GCD.

`gcloud.environment_vars.GCD_HOST = 'DATASTORE_HOST'`

Environment variable defining host for GCD dataset server.

`gcloud.environment_vars.PROJECT = 'GLOUD_PROJECT'`

Environment variable defining default project.

`gcloud.environment_vars.PUBSUB_EMULATOR = 'PUBSUB_EMULATOR_HOST'`

Environment variable defining host for Pub/Sub emulator.

`gcloud.environment_vars.TESTS_DATASET = 'GLOUD_TESTS_DATASET_ID'`

Environment variable defining dataset ID for tests.

`gcloud.environment_vars.TESTS_PROJECT = 'GLOUD_TESTS_PROJECT_ID'`

Environment variable defining project for tests.

Authentication

6.1 Overview

- If you're running in **Compute Engine or App Engine**, authentication should “just work”.
- If you're developing locally, the easiest way to authenticate is using the [Google Cloud SDK](#):

```
$ gcloud auth login
```

- If you're running your application elsewhere, you should download a [service account](#) JSON keyfile and point to it using an environment variable:

```
$ export GOOGLE_APPLICATION_CREDENTIALS="/path/to/keyfile.json"
```

6.2 Client-Provided Authentication

Every package uses a *Client* as a base for interacting with an API. For example:

```
from gcloud import datastore
client = datastore.Client()
```

Passing no arguments at all will “just work” if you've followed the instructions in the [Overview](#). The credentials are inferred from your local environment by using [Google Application Default Credentials](#).

6.2.1 Credential Discovery Precedence

When loading the [Application Default Credentials](#), the library will check properties of your local environment in the following order:

1. Application running in Google App Engine
2. JSON or PKCS12/P12 keyfile pointed to by `GOOGLE_APPLICATION_CREDENTIALS` environment variable
3. Credentials provided by the Google Cloud SDK (via `gcloud auth login`)
4. Application running in Google Compute Engine

6.3 Explicit Credentials

The Application Default Credentials discussed above can be useful if your code needs to run in many different environments or if you just don't want authentication to be a focus in your code.

However, you may want to be explicit because

- your code will only run in one place
- you may have code which needs to be run as a specific service account every time (rather than with the locally inferred credentials)
- you may want to use two separate accounts to simultaneously access data from different projects

In these situations, you can create an explicit `Credentials` object suited to your environment. After creation, you can pass it directly to a `Client`:

```
client = Client(credentials=credentials)
```

6.3.1 Google App Engine Environment

To create `credentials` just for Google App Engine:

```
from oauth2client.appengine import AppAssertionCredentials
credentials = AppAssertionCredentials([])
```

6.3.2 Google Compute Engine Environment

To create `credentials` just for Google Compute Engine:

```
from oauth2client.gce import AppAssertionCredentials
credentials = AppAssertionCredentials([])
```

6.3.3 Service Accounts

A `service account` can be used with both a JSON keyfile and a PKCS12/P12 keyfile.

Directly creating `credentials` in `oauth2client` for a service account is a rather complex process, so as a convenience, the `from_service_account_json()` and `from_service_account_p12()` factories are provided to create a `Client` with service account credentials.

For example, with a JSON keyfile:

```
client = Client.from_service_account_json('/path/to/keyfile.json')
```

Tip: Unless you have a specific reason to use a PKCS12/P12 key for your service account, we recommend using a JSON key.

6.3.4 User Accounts (3-legged OAuth 2.0) with a refresh token

The majority of cases are intended to authenticate machines or workers rather than actual user accounts. However, it's also possible to call Google Cloud APIs with a user account via [OAuth 2.0](#).

Tip: A production application should **use a service account**, but you may wish to use your own personal user account when first getting started with the `gcloud-python` library.

The simplest way to use credentials from a user account is via Application Default Credentials using `gcloud auth login` (as mentioned above):

```
from oauth2client.client import GoogleCredentials
credentials = GoogleCredentials.get_application_default()
```

This will still follow the *precedence* described above, so be sure none of the other possible environments conflict with your user provided credentials.

Advanced users of `oauth2client` can also use custom flows to create credentials using `client secrets` or using a `webserver flow`. After creation, `Credentials` can be serialized with `to_json()` and stored in a file and then and deserialized with `from_json()`.

6.4 Troubleshooting

6.4.1 Setting up a Service Account

If your application is not running on Google Compute Engine, you need a [Google Developers Service Account](#).

1. Visit the [Google Developers Console](#).
2. Create a new project or click on an existing project.
3. Navigate to **APIs & auth > APIs** and enable the APIs that your application requires.

Note: You may need to enable billing in order to use these services.

- **BigQuery**
 - BigQuery API
 - **Datastore**
 - Google Cloud Datastore API
 - **Pub/Sub**
 - Google Cloud Pub/Sub
 - **Search**
 - Google Cloud Search API
 - **Storage**
 - Google Cloud Storage
 - Google Cloud Storage JSON API
-

1. Navigate to **APIs & auth > Credentials**.

You should see a screen like one of the following:

Find the “Add credentials” drop down and select “Service account” to be guided through downloading a new JSON keyfile.

If you want to re-use an existing service account, you can easily generate a new keyfile. Just select the account you wish to re-use, and click **Generate new JSON key**:

6.4.2 Using Google Compute Engine

If your code is running on Google Compute Engine, using the inferred Google [Application Default Credentials](#) will be sufficient for retrieving credentials.

However, by default your credentials may not grant you access to the services you intend to use. Be sure when you [set up the GCE instance](#), you add the correct scopes for the APIs you want to access:

- **All APIs**
 - `https://www.googleapis.com/auth/cloud-platform`
 - `https://www.googleapis.com/auth/cloud-platform.read-only`
- **BigQuery**
 - `https://www.googleapis.com/auth/bigquery`
 - `https://www.googleapis.com/auth/bigquery.insertdata`
- **Datastore**
 - `https://www.googleapis.com/auth/datastore`
 - `https://www.googleapis.com/auth/userinfo.email`
- **Pub/Sub**
 - `https://www.googleapis.com/auth/pubsub`
- **Storage**
 - `https://www.googleapis.com/auth/devstorage.full_control`
 - `https://www.googleapis.com/auth/devstorage.read_only`
 - `https://www.googleapis.com/auth/devstorage.read_write`

6.5 Advanced Customization

Though the `gcloud-python` library defaults to using `oauth2client` to sign requests and `httplib2` for sending requests, it is not a strict requirement.

The `Client` constructor accepts an optional `http` argument in place of a `credentials` object. If passed, all HTTP requests made by the client will use your custom HTTP object.

In order for this to be possible, the `http` object must do two things:

- Handle authentication on its own
- Define a method `request()` that can substitute for `httplib2.Http.request()`.

The entire signature from `httplib2` need not be implemented, we only use it as

```
http.request(uri, method=method_name, body=body, headers=headers)
```

For an example of such an implementation, a `gcloud-python` user created a [custom HTTP class](#) using the `requests` library.

As for handling authentication on your own, it may be easiest just to re-use bits from `oauth2client`. Unfortunately, these parts have a hard dependency on `httplib2`. We hope to enable using [custom HTTP libraries](#) with `oauth2client` at some point.

Datastore Client

Convenience wrapper for invoking APIs/factories w/ a project.

class `gcloud.datastore.client.Client` (*project=None, namespace=None, credentials=None, http=None*)

Bases: `gcloud.client.Client`

Convenience wrapper for invoking APIs/factories w/ a project.

Parameters

- **project** (*string*) – (optional) The project to pass to proxied API methods.
- **namespace** (*string*) – (optional) namespace to pass to proxied API methods.
- **credentials** (`oauth2client.client.OAuth2Credentials` or `NoneType`) – The OAuth2 Credentials to use for the connection owned by this client. If not passed (and if no `http` object is passed), falls back to the default inferred from the environment.
- **http** (`httplib2.Http` or class that defines `request()`) – An optional HTTP object to make requests. If not passed, an `http` object is created that is bound to the `credentials` for the current object.

allocate_ids (*incomplete_key, num_ids*)

Allocate a list of IDs from a partial key.

Parameters

- **incomplete_key** (`gcloud.datastore.key.Key`) – Partial key to use as base for allocated IDs.
- **num_ids** (*int*) – The number of IDs to allocate.

Return type list of `gcloud.datastore.key.Key`

Returns The (complete) keys allocated with `incomplete_key` as root.

Raises `ValueError` if `incomplete_key` is not a partial key.

batch ()

Proxy to `gcloud.datastore.batch.Batch`.

current_batch

Currently-active batch.

Return type `gcloud.datastore.batch.Batch`, or an object implementing its API, or `NoneType` (if no batch is active).

Returns The batch/transaction at the top of the batch stack.

current_transaction

Currently-active transaction.

Return type `gcloud.datastore.transaction.Transaction`, or an object implementing its API, or `NoneType` (if no transaction is active).

Returns The transaction at the top of the batch stack.

delete (*key*)

Delete the key in the Cloud Datastore.

Note: This is just a thin wrapper over `delete_multi()`. The backend API does not make a distinction between a single key or multiple keys in a commit request.

Parameters **key** (`gcloud.datastore.key.Key`) – The key to be deleted from the datastore.

delete_multi (*keys*)

Delete keys from the Cloud Datastore.

Parameters **keys** (list of `gcloud.datastore.key.Key`) – The keys to be deleted from the datastore.

get (*key, missing=None, deferred=None*)

Retrieve an entity from a single key (if it exists).

Note: This is just a thin wrapper over `get_multi()`. The backend API does not make a distinction between a single key or multiple keys in a lookup request.

Parameters

- **key** (`gcloud.datastore.key.Key`) – The key to be retrieved from the datastore.
- **missing** (*list*) – (Optional) If a list is passed, the key-only entities returned by the backend as “missing” will be copied into it.
- **deferred** (*list*) – (Optional) If a list is passed, the keys returned by the backend as “deferred” will be copied into it.

Return type `gcloud.datastore.entity.Entity` or `NoneType`

Returns The requested entity if it exists.

get_multi (*keys, missing=None, deferred=None*)

Retrieve entities, along with their attributes.

Parameters

- **keys** (list of `gcloud.datastore.key.Key`) – The keys to be retrieved from the datastore.
- **missing** (*list*) – (Optional) If a list is passed, the key-only entities returned by the backend as “missing” will be copied into it. If the list is not empty, an error will occur.
- **deferred** (*list*) – (Optional) If a list is passed, the keys returned by the backend as “deferred” will be copied into it. If the list is not empty, an error will occur.

Return type list of `gcloud.datastore.entity.Entity`

Returns The requested entities.

Raises `ValueError` if one or more of `keys` has a project which does not match our project.

key (**path_args*, ***kwargs*)

Proxy to `gcloud.datastore.key.Key`.

Passes our project.

put (*entity*)

Save an entity in the Cloud Datastore.

Note: This is just a thin wrapper over `put_multi()`. The backend API does not make a distinction between a single entity or multiple entities in a commit request.

Parameters **entity** (`gcloud.datastore.entity.Entity`) – The entity to be saved to the datastore.

put_multi (*entities*)

Save entities in the Cloud Datastore.

Parameters **entities** (list of `gcloud.datastore.entity.Entity`) – The entities to be saved to the datastore.

Raises `ValueError` if `entities` is a single entity.

query (***kwargs*)

Proxy to `gcloud.datastore.query.Query`.

Passes our project.

transaction ()

Proxy to `gcloud.datastore.transaction.Transaction`.

7.1 Connection

Connections to gcloud datastore API servers.

class `gcloud.datastore.connection.Connection` (*credentials=None*, *http=None*,
api_base_url=None)

Bases: `gcloud.connection.Connection`

A connection to the Google Cloud Datastore via the Protobuf API.

This class should understand only the basic types (and protobufs) in method arguments, however should be capable of returning advanced types.

Parameters

- **credentials** (`oauth2client.client.OAuth2Credentials`) – The OAuth2 Credentials to use for this connection.
- **http** (`httplib2.Http` or class that defines `request()`) – An optional HTTP object to make requests.
- **api_base_url** (*string*) – The base of the API call URL. Defaults to `API_BASE_URL`.

API_BASE_URL = 'https://www.googleapis.com'

The base of the API call URL.

API_URL_TEMPLATE = '{api_base}/datastore/{api_version}/datasets/{project}/{method}'

A template for the URL of a particular API call.

API_VERSION = 'v1beta2'

The version of the API, used in building the API call's URL.

SCOPE = ('https://www.googleapis.com/auth/datastore', 'https://www.googleapis.com/auth/userinfo.email')

The scopes required for authenticating as a Cloud Datastore consumer.

allocate_ids (*project*, *key_pbs*)

Obtain backend-generated IDs for a set of keys.

Maps the `DatastoreService.AllocateIds` protobuf RPC.

Parameters

- **project** (*string*) – The project to which the transaction belongs.
- **key_pbs** (list of `gcloud.datastore._generated.entity_pb2.Key`) – The keys for which the backend should allocate IDs.

Return type list of `gcloud.datastore._generated.entity_pb2.Key`

Returns An equal number of keys, with IDs filled in by the backend.

begin_transaction (*project*)

Begin a transaction.

Maps the `DatastoreService.BeginTransaction` protobuf RPC.

Parameters **project** (*string*) – The project to which the transaction applies.

Return type bytes

Returns The serialized transaction that was begun.

build_api_url (*project*, *method*, *base_url=None*, *api_version=None*)

Construct the URL for a particular API call.

This method is used internally to come up with the URL to use when making RPCs to the Cloud Datastore API.

Parameters

- **project** (*string*) – The project to connect to. This is usually your project name in the cloud console.
- **method** (*string*) – The API method to call (e.g. 'runQuery', 'lookup').
- **base_url** (*string*) – The base URL where the API lives. You shouldn't have to provide this.
- **api_version** (*string*) – The version of the API to connect to. You shouldn't have to provide this.

commit (*project*, *request*, *transaction_id*)

Commit mutations in context of current transaction (if any).

Maps the `DatastoreService.Commit` protobuf RPC.

Parameters

- **project** (*string*) – The project to which the transaction applies.

- **request** (`_generated.datastore_pb2.CommitRequest`) – The protobuf with the mutations being committed.
- **transaction_id** (*string or None*) – The transaction ID returned from `begin_transaction()`. Non-transactional batches must pass None.

Note: This method will mutate `request` before using it.

Return type `tuple`

Returns' The pair of the number of index updates and a list of `_generated.entity_pb2.Key` for each incomplete key that was completed in the commit.

lookup (*project, key_pbs, eventual=False, transaction_id=None*)

Lookup keys from a project in the Cloud Datastore.

Maps the `DatastoreService.Lookup` protobuf RPC.

This uses mostly protobufs (`gcloud.datastore._generated.entity_pb2.Key` as input and `gcloud.datastore._generated.entity_pb2.Entity` as output). It is used under the hood in `Client.get()`:

```
>>> from gcloud import datastore
>>> client = datastore.Client(project='project')
>>> key = client.key('MyKind', 1234)
>>> client.get(key)
[<Entity object>]
```

Using a `Connection` directly:

```
>>> connection.lookup('project', [key.to_protobuf()])
[<Entity protobuf>]
```

Parameters

- **project** (*string*) – The project to look up the keys in.
- **key_pbs** (list of `gcloud.datastore._generated.entity_pb2.Key`) – The keys to retrieve from the datastore.
- **eventual** (*bool*) – If False (the default), request STRONG read consistency. If True, request EVENTUAL read consistency.
- **transaction_id** (*string*) – If passed, make the request in the scope of the given transaction. Incompatible with `eventual==True`.

Return type `tuple`

Returns A triple of (results, missing, deferred) where both results and missing are lists of `gcloud.datastore._generated.entity_pb2.Entity` and deferred is a list of `gcloud.datastore._generated.entity_pb2.Key`.

rollback (*project, transaction_id*)

Rollback the connection's existing transaction.

Maps the `DatastoreService.Rollback` protobuf RPC.

Parameters

- **project** (*string*) – The project to which the transaction belongs.
- **transaction_id** (*string*) – The transaction ID returned from `begin_transaction()`.

run_query (*project, query_pb, namespace=None, eventual=False, transaction_id=None*)

Run a query on the Cloud Datastore.

Maps the `DatastoreService.RunQuery` protobuf RPC.

Given a `Query` protobuf, sends a `runQuery` request to the Cloud Datastore API and returns a list of entity protobufs matching the query.

You typically wouldn't use this method directly, in favor of the `gcloud.datastore.query.Query.fetch()` method.

Under the hood, the `gcloud.datastore.query.Query` class uses this method to fetch data:

```
>>> from gcloud import datastore
>>> query = datastore.Query(kind='MyKind')
>>> query.add_filter('property', '=', 'val')
```

Using the query iterator's `next_page()` method:

```
>>> query_iter = query.fetch()
>>> entities, more_results, cursor = query_iter.next_page()
>>> entities
[<list of Entity unmarshalled from protobuf>]
>>> more_results
<boolean of more results>
>>> cursor
<string containing cursor where fetch stopped>
```

Under the hood this is doing:

```
>>> connection.run_query('project', query.to_protobuf())
[<list of Entity Protobufs>], cursor, more_results, skipped_results
```

Parameters

- **project** (*string*) – The project over which to run the query.
- **query_pb** (`gcloud.datastore._generated.query_pb2.Query`) – The Protobuf representing the query to run.
- **namespace** (*string*) – The namespace over which to run the query.
- **eventual** (*bool*) – If `False` (the default), request `STRONG` read consistency. If `True`, request `EVENTUAL` read consistency.
- **transaction_id** (*string*) – If passed, make the request in the scope of the given transaction. Incompatible with `eventual==True`.

Entities

Class for representing a single entity in the Cloud Datastore.

```
class gcloud.datastore.entity.Entity (key=None, exclude_from_indexes=())
    Bases: dict
```

Entities are akin to rows in a relational database

An entity storing the actual instance of data.

Each entity is officially represented with a `gcloud.datastore.key.Key` class, however it is possible that you might create an Entity with only a partial Key (that is, a Key with a Kind, and possibly a parent, but without an ID). In such a case, the datastore service will automatically assign an ID to the partial key.

Entities in this API act like dictionaries with extras built in that allow you to delete or persist the data stored on the entity.

Entities are mutable and act like a subclass of a dictionary. This means you could take an existing entity and change the key to duplicate the object.

Use `gcloud.datastore.get()` to retrieve an existing entity.

```
>>> datastore.get(key)
<Entity[{'kind': 'EntityKind', id: 1234}] {'property': 'value'}>
```

You can the set values on the entity just like you would on any other dictionary.

```
>>> entity['age'] = 20
>>> entity['name'] = 'JJ'
>>> entity
<Entity[{'kind': 'EntityKind', id: 1234}] {'age': 20, 'name': 'JJ'}>
```

And you can convert an entity to a regular Python dictionary with the `dict` builtin:

```
>>> dict(entity)
{'age': 20, 'name': 'JJ'}
```

Note: When saving an entity to the backend, values which are “text” (unicode in Python2, str in Python3) will be saved using the ‘text_value’ field, after being encoded to UTF-8. When retrieved from the back-end, such values will be decoded to “text” again. Values which are “bytes” (str in Python2, bytes in Python3), will be saved using the ‘blob_value’ field, without any decoding / encoding step.

Parameters

- **key** (*gcloud.datastore.key.Key*) – Optional key to be set on entity. Required for `gcloud.datastore.put()` and `gcloud.datastore.put_multi()`
- **exclude_from_indexes** (*tuple of string*) – Names of fields whose values are not to be indexed for this entity.

exclude_from_indexes

Names of fields which are *not* to be indexed for this entity.

Return type sequence of field names

kind

Get the kind of the current entity.

Note: This relies entirely on the *gcloud.datastore.key.Key* set on the entity. That means that we're not storing the kind of the entity at all, just the properties and a pointer to a Key which knows its Kind.

Keys

Create / interact with gcloud datastore keys.

```
class gcloud.datastore.key.Key(*path_args,**kwargs)
    Bases: object
```

An immutable representation of a datastore Key.

To create a basic key:

```
>>> Key('EntityKind', 1234)
<Key[{'kind': 'EntityKind', 'id': 1234}]>
>>> Key('EntityKind', 'foo')
<Key[{'kind': 'EntityKind', 'name': 'foo'}]>
```

To create a key with a parent:

```
>>> Key('Parent', 'foo', 'Child', 1234)
<Key[{'kind': 'Parent', 'name': 'foo'}, {'kind': 'Child', 'id': 1234}]>
>>> Key('Child', 1234, parent=parent_key)
<Key[{'kind': 'Parent', 'name': 'foo'}, {'kind': 'Child', 'id': 1234}]>
```

To create a partial key:

```
>>> Key('Parent', 'foo', 'Child')
<Key[{'kind': 'Parent', 'name': 'foo'}, {'kind': 'Child'}]>
```

Parameters

- **path_args** (*tuple of string and integer*) – May represent a partial (odd length) or full (even length) key path.
- **kwargs** (*dict*) – Keyword arguments to be passed in.

Accepted keyword arguments are

- **namespace** (string): A namespace identifier for the key.
- **project** (string): The project associated with the key.
- **parent** (*gcloud.datastore.key.Key*): The parent of the key.

The project argument is required unless it has been set implicitly.

completed_key (*id_or_name*)

Creates new key from existing partial key by adding final ID/name.

Parameters **id_or_name** (*string or integer*) – ID or name to be added to the key.

Return type *gcloud.datastore.key.Key*

Returns A new `Key` instance with the same data as the current one and an extra ID or name added.

Raises `ValueError` if the current key is not partial or if `id_or_name` is not a string or integer.

flat_path

Getter for the key path as a tuple.

Return type tuple of string and integer

Returns The tuple of elements in the path.

id

ID getter. Based on the last element of path.

Return type integer

Returns The (integer) ID of the key.

id_or_name

Getter. Based on the last element of path.

Return type integer (if `id`) or string (if `name`)

Returns The last element of the key's path if it is either an `id` or a `name`.

is_partial

Boolean indicating if the key has an ID (or name).

Return type `bool`

Returns `True` if the last element of the key's path does not have an `id` or a `name`.

kind

Kind getter. Based on the last element of path.

Return type `string`

Returns The kind of the current key.

name

Name getter. Based on the last element of path.

Return type `string`

Returns The (string) name of the key.

namespace

Namespace getter.

Return type `string`

Returns The namespace of the current key.

parent

The parent of the current key.

Return type *gcloud.datastore.key.Key* or `NoneType`

Returns A new `Key` instance, whose path consists of all but the last element of current path. If the current key has only one path element, returns `None`.

path

Path getter.

Returns a copy so that the key remains immutable.

Return type `list of dict`

Returns The (key) path of the current key.

project

Project getter.

Return type `string`

Returns The key's project.

to_protobuf()

Return a protobuf corresponding to the key.

Return type `gcloud.datastore._generated.entity_pb2.Key`

Returns The protobuf representing the key.

Queries

Create / interact with gcloud datastore queries.

```
class gcloud.datastore.query.Iterator (query, client, limit=None, offset=0, start_cursor=None, end_cursor=None)
```

Bases: `object`

Represent the state of a given execution of a Query.

Parameters

- **query** (*gcloud.datastore.query.Query*) – Query object holding permanent configuration (i.e. things that don't change on with each page in a results set).
- **client** (*gcloud.datastore.client.Client*) – The client used to make a request.
- **limit** (*integer*) – (Optional) Limit the number of results returned.
- **offset** (*integer*) – (Optional) Defaults to 0. Offset used to begin a query.
- **start_cursor** (*bytes*) – (Optional) Cursor to begin paging through query results.
- **end_cursor** (*bytes*) – (Optional) Cursor to end paging through query results.

next_page ()

Fetch a single “page” of query results.

Low-level API for fine control: the more convenient API is to iterate on the current Iterator.

Return type tuple, (entities, more_results, cursor)

```
class gcloud.datastore.query.Query (client, kind=None, project=None, namespace=None, ancestor=None, filters=(), projection=(), order=(), group_by=())
```

Bases: `object`

A Query against the Cloud Datastore.

This class serves as an abstraction for creating a query over data stored in the Cloud Datastore.

Parameters

- **client** (*gcloud.datastore.client.Client*) – The client used to connect to datastore.
- **kind** (*string*) – The kind to query.
- **project** (*string*) – The project associated with the query. If not passed, uses the client's value.

- **namespace** (*string* or *None*) – The namespace to which to restrict results. If not passed, uses the client’s value.
- **ancestor** (*gcloud.datastore.key.Key* or *None*) – key of the ancestor to which this query’s results are restricted.
- **filters** (*sequence of (property_name, operator, value) tuples*) – property filters applied by this query.
- **projection** (*sequence of string*) – fields returned as part of query results.
- **order** (*sequence of string*) – field names used to order query results. Prepend ‘-’ to a field name to sort it in descending order.
- **group_by** (*sequence of string*) – field names used to group query results.

Raises `ValueError` if `project` is not passed and no implicit default is set.

OPERATORS = {'>': 3, '<=': 2, '=': 5, '>=': 4, '<': 1}

Mapping of operator strings and their protobuf equivalents.

add_filter (*property_name, operator, value*)

Filter the query based on a property name, operator and a value.

Expressions take the form of:

```
.add_filter('<property>', '<operator>', <value>)
```

where `property` is a property stored on the entity in the datastore and `operator` is one of `OPERATORS` (ie, =, <, <=, >, >=):

```
>>> from gcloud import datastore
>>> query = datastore.Query('Person')
>>> query.add_filter('name', '=', 'James')
>>> query.add_filter('age', '>', 50)
```

Parameters

- **property_name** (*string*) – A property name.
- **operator** (*string*) – One of =, <, <=, >, >=.
- **value** (*int, str, bool, float, NoneType, :class'datetime.datetime'*) – The value to filter on.

Raises `ValueError` if `operation` is not one of the specified values, or if a filter names ‘`__key__`’ but passes an invalid value (a key is required).

ancestor

The ancestor key for the query.

Return type `Key` or `None`

fetch (*limit=None, offset=0, start_cursor=None, end_cursor=None, client=None*)

Execute the Query; return an iterator for the matching entities.

For example:

```
>>> from gcloud import datastore
>>> query = datastore.Query('Person')
>>> query.add_filter('name', '=', 'Sally')
>>> list(query.fetch())
[<Entity object>, <Entity object>, ...]
```

```
>>> list(query.fetch(1))
[<Entity object>]
```

Parameters

- **limit** (*integer or None*) – An optional limit passed through to the iterator.
- **offset** (*integer*) – An optional offset passed through to the iterator.
- **start_cursor** (*bytes*) – An optional cursor passed through to the iterator.
- **end_cursor** (*bytes*) – An optional cursor passed through to the iterator.
- **client** (*gcloud.datastore.client.Client*) – client used to connect to datastore. If not supplied, uses the query's value.

Return type *Iterator***Raises** `ValueError` if `connection` is not passed and no implicit default has been set.**filters**

Filters set on the query.

Return type sequence of (property_name, operator, value) tuples.**group_by**

Names of fields used to group query results.

Return type sequence of string**key_filter** (*key, operator='='*)

Filter on a key.

Parameters

- **key** (*gcloud.datastore.key.Key*) – The key to filter on.
- **operator** (*string*) – (Optional) One of =, <, <=, >, >=. Defaults to =.

keys_only ()

Set the projection to include only keys.

kind

Get the Kind of the Query.

Return type *string***namespace**

This query's namespace

Return type *string or None***Returns** the namespace assigned to this query**order**

Names of fields used to sort query results.

Return type sequence of string**project**

Get the project for this Query.

Return type *str***projection**

Fields names returned by the query.

Return type sequence of string

Returns Names of fields in query results.

Transactions

Create / interact with gcloud datastore transactions.

```
class gcloud.datastore.transaction.Transaction (client)
    Bases: gcloud.datastore.batch.Batch
```

An abstraction representing datastore Transactions.

Transactions can be used to build up a bulk mutation and ensure all or none succeed (transactionally).

For example, the following snippet of code will put the two save operations (either `insert_auto_id` or `upsert`) into the same mutation, and execute those within a transaction:

```
>>> from gcloud import datastore

>>> with datastore.Transaction():
...     datastore.put_multi([entity1, entity2])
```

Because it derives from *Batch*, *Transaction* also provides *put()* and *delete()* methods:

```
>>> with datastore.Transaction() as xact:
...     xact.put(entity1)
...     xact.delete(entity2.key)
```

By default, the transaction is rolled back if the transaction block exits with an error:

```
>>> with datastore.Transaction():
...     do_some_work()
...     raise SomeException() # rolls back
```

If the transaction block exists without an exception, it will commit by default.

Warning: Inside a transaction, automatically assigned IDs for entities will not be available at save time! That means, if you try:

```
>>> with datastore.Transaction():
...     entity = datastore.Entity(key=Key('Thing'))
...     datastore.put(entity)
```

entity won't have a complete Key until the transaction is committed. Once you exit the transaction (or call `commit()`), the automatically generated ID will be assigned to the entity:

```
>>> with datastore.Transaction():
...     entity = datastore.Entity(key=Key('Thing'))
...     datastore.put(entity)
...     print entity.key.is_partial # There is no ID on this key.
...
True
>>> print entity.key.is_partial # There is an ID.
False
```

If you don't want to use the context manager you can initialize a transaction manually:

```
>>> transaction = datastore.Transaction()
>>> transaction.begin()

>>> entity = datastore.Entity(key=Key('Thing'))
>>> transaction.put(entity)

>>> if error:
...     transaction.rollback()
... else:
...     transaction.commit()
```

Parameters `client` (`gcloud.datastore.client.Client`) – the client used to connect to datastore.

`begin()`

Begins a transaction.

This method is called automatically when entering a `with` statement, however it can be called explicitly if you don't want to use a context manager.

Raises `ValueError` if the transaction has already begun.

`commit()`

Commits the transaction.

This is called automatically upon exiting a `with` statement, however it can be called explicitly if you don't want to use a context manager.

This method has necessary side-effects:

- Sets the current transaction's ID to `None`.

`connection`

Getter for connection over which the batch will run.

Return type `gcloud.datastore.connection.Connection`

Returns The connection over which the batch will run.

current ()

Return the topmost transaction.

Note: If the topmost element on the stack is not a transaction, returns None.

Return type `gcloud.datastore.transaction.Transaction` or None

delete (key)

Remember a key to be deleted during `commit ()`.

Parameters **key** (`gcloud.datastore.key.Key`) – the key to be deleted.

Raises `ValueError` if key is not complete, or if the key's `project` does not match ours.

id

Getter for the transaction ID.

Return type `string`

Returns The ID of the current transaction.

mutations

Getter for the changes accumulated by this batch.

Every batch is committed with a single commit request containing all the work to be done as mutations. Inside a batch, calling `put ()` with an entity, or `delete ()` with a key, builds up the request by adding a new mutation. This getter returns the protobuf that has been built-up so far.

Return type `gcloud.datastore._generated.datastore_pb2.Mutation`

Returns The Mutation protobuf to be sent in the commit request.

namespace

Getter for namespace in which the batch will run.

Return type `str`

Returns The namespace in which the batch will run.

project

Getter for project in which the batch will run.

Return type `str`

Returns The project in which the batch will run.

put (entity)

Remember an entity's state to be saved during `commit ()`.

Note: Any existing properties for the entity will be replaced by those currently set on this instance. Already-stored properties which do not correspond to keys set on this instance will be removed from the datastore.

Note: Property values which are “text” (‘unicode’ in Python2, ‘str’ in Python3) map to ‘string_value’ in the datastore; values which are “bytes” (‘str’ in Python2, ‘bytes’ in Python3) map to ‘blob_value’.

When an entity has a partial key, calling `commit()` sends it as an `insert_auto_id` mutation and the key is completed. On return, the key for the `entity` passed in is updated to match the key ID assigned by the server.

Parameters `entity` (`gcloud.datastore.entity.Entity`) – the entity to be saved.

Raises `ValueError` if entity has no key assigned, or if the key's `project` does not match ours.

rollback()

Rolls back the current transaction.

This method has necessary side-effects:

- Sets the current connection's transaction reference to `None`.
- Sets the current transaction's ID to `None`.

Batches

Create / interact with a batch of updates / deletes.

Batches provide the ability to execute multiple operations in a single request to the Cloud Datastore API.

See https://cloud.google.com/datastore/docs/concepts/entities#Datastore_Batch_operations

class `gcloud.datastore.batch.Batch` (*client*)

Bases: `object`

An abstraction representing a collected group of updates / deletes.

Used to build up a bulk mutation.

For example, the following snippet of code will put the two `save` operations and the `delete` operation into the same mutation, and send them to the server in a single API request:

```
>>> from gcloud.datastore.batch import Batch
>>> batch = Batch()
>>> batch.put(entity1)
>>> batch.put(entity2)
>>> batch.delete(key3)
>>> batch.commit()
```

You can also use a batch as a context manager, in which case `commit()` will be called automatically if its block exits without raising an exception:

```
>>> with Batch() as batch:
...     batch.put(entity1)
...     batch.put(entity2)
...     batch.delete(key3)
```

By default, no updates will be sent if the block exits with an error:

```
>>> with Batch() as batch:
...     do_some_work(batch)
...     raise Exception() # rolls back
```

Parameters `client` (`gcloud.datastore.client.Client`) – The client used to connect to datastore.

begin()

Begins a batch.

This method is called automatically when entering a `with` statement, however it can be called explicitly if you don't want to use a context manager.

Overridden by `gcloud.datastore.transaction.Transaction`.

Raises `ValueError` if the batch has already begun.

commit ()

Commits the batch.

This is called automatically upon exiting a `with` statement, however it can be called explicitly if you don't want to use a context manager.

connection

Getter for connection over which the batch will run.

Return type `gcloud.datastore.connection.Connection`

Returns The connection over which the batch will run.

current ()

Return the topmost batch / transaction, or `None`.

delete (key)

Remember a key to be deleted during `commit ()`.

Parameters **key** (`gcloud.datastore.key.Key`) – the key to be deleted.

Raises `ValueError` if key is not complete, or if the key's `project` does not match ours.

mutations

Getter for the changes accumulated by this batch.

Every batch is committed with a single `commit` request containing all the work to be done as mutations. Inside a batch, calling `put ()` with an entity, or `delete ()` with a key, builds up the request by adding a new mutation. This getter returns the protobuf that has been built-up so far.

Return type `gcloud.datastore._generated.datastore_pb2.Mutation`

Returns The Mutation protobuf to be sent in the `commit` request.

namespace

Getter for namespace in which the batch will run.

Return type `str`

Returns The namespace in which the batch will run.

project

Getter for project in which the batch will run.

Return type `str`

Returns The project in which the batch will run.

put (entity)

Remember an entity's state to be saved during `commit ()`.

Note: Any existing properties for the entity will be replaced by those currently set on this instance. Already-stored properties which do not correspond to keys set on this instance will be removed from the datastore.

Note: Property values which are "text" ('unicode' in Python2, 'str' in Python3) map to 'string_value' in the datastore; values which are "bytes" ('str' in Python2, 'bytes' in Python3) map to 'blob_value'.

When an entity has a partial key, calling `commit()` sends it as an `insert_auto_id` mutation and the key is completed. On return, the key for the `entity` passed in is updated to match the key ID assigned by the server.

Parameters `entity` (`gcloud.datastore.entity.Entity`) – the entity to be saved.

Raises `ValueError` if entity has no key assigned, or if the key's `project` does not match ours.

rollback()

Rolls back the current batch.

Marks the batch as aborted (can't be used again).

Overridden by `gcloud.datastore.transaction.Transaction`.

Helpers

Helper functions for dealing with Cloud Datastore's Protobuf API.

The non-private functions are part of the API.

`gcloud.datastore.helpers.entity_from_protobuf(pb)`

Factory method for creating an entity based on a protobuf.

The protobuf should be one returned from the Cloud Datastore Protobuf API.

Parameters `pb` (`gcloud.datastore._generated.entity_pb2.Entity`) – The Protobuf representing the entity.

Return type `gcloud.datastore.entity.Entity`

Returns The entity derived from the protobuf.

`gcloud.datastore.helpers.key_from_protobuf(pb)`

Factory method for creating a key based on a protobuf.

The protobuf should be one returned from the Cloud Datastore Protobuf API.

Parameters `pb` (`gcloud.datastore._generated.entity_pb2.Key`) – The Protobuf representing the key.

Return type `gcloud.datastore.key.Key`

Returns a new `Key` instance

Storage Client

Client for interacting with the Google Cloud Storage API.

class `gcloud.storage.client.Client` (*project=None, credentials=None, http=None*)
 Bases: `gcloud.client.JSONClient`

Client to bundle configuration needed for API requests.

Parameters

- **project** (*string*) – the project which the client acts on behalf of. Will be passed when creating a topic. If not passed, falls back to the default inferred from the environment.
- **credentials** (`oauth2client.client.OAuth2Credentials` or `NoneType`) – The OAuth2 Credentials to use for the connection owned by this client. If not passed (and if no `http` object is passed), falls back to the default inferred from the environment.
- **http** (`httplib2.Http` or class that defines `request()`) – An optional HTTP object to make requests. If not passed, an `http` object is created that is bound to the `credentials` for the current object.

`batch()`

Factory constructor for batch object.

Note: This will not make an HTTP request; it simply instantiates a batch object owned by this client.

Return type `gcloud.storage.batch.Batch`

Returns The batch object created.

`bucket(bucket_name)`

Factory constructor for bucket object.

Note: This will not make an HTTP request; it simply instantiates a bucket object owned by this client.

Parameters **bucket_name** (*string*) – The name of the bucket to be instantiated.

Return type `gcloud.storage.bucket.Bucket`

Returns The bucket object created.

connection

Get connection or batch on the client.

Return type *gcloud.storage.connection.Connection*

Returns The connection set on the client, or the batch if one is set.

create_bucket (*bucket_name*)

Create a new bucket.

For example:

```
>>> bucket = client.create_bucket('my-bucket')
>>> print bucket
<Bucket: my-bucket>
```

This implements “storage.buckets.insert”.

If the bucket already exists, will raise *gcloud.exceptions.Conflict*.

Parameters **bucket_name** (*string*) – The bucket name to create.

Return type *gcloud.storage.bucket.Bucket*

Returns The newly created bucket.

current_batch

Currently-active batch.

Return type *gcloud.storage.batch.Batch* or *NoneType* (if no batch is active).

Returns The batch at the top of the batch stack.

get_bucket (*bucket_name*)

Get a bucket by name.

If the bucket isn’t found, this will raise a *gcloud.storage.exceptions.NotFound*.

For example:

```
>>> try:
>>>     bucket = client.get_bucket('my-bucket')
>>> except gcloud.exceptions.NotFound:
>>>     print 'Sorry, that bucket does not exist!'
```

This implements “storage.buckets.get”.

Parameters **bucket_name** (*string*) – The name of the bucket to get.

Return type *gcloud.storage.bucket.Bucket*

Returns The bucket matching the name provided.

Raises *gcloud.exceptions.NotFound*

list_buckets (*max_results=None, page_token=None, prefix=None, projection='noAcl', fields=None*)

Get all buckets in the project associated to the client.

This will not populate the list of blobs available in each bucket.

```
>>> for bucket in client.list_buckets():
>>>     print bucket
```

This implements “storage.buckets.list”.

Parameters

- **max_results** (integer or `NoneType`) – Optional. Maximum number of buckets to return.
- **page_token** (string or `NoneType`) – Optional. Opaque marker for the next “page” of buckets. If not passed, will return the first page of buckets.
- **prefix** (string or `NoneType`) – Optional. Filter results to buckets whose names begin with this prefix.
- **projection** (string or `NoneType`) – If used, must be ‘full’ or ‘noAcl’. Defaults to ‘noAcl’. Specifies the set of properties to return.
- **fields** (string or `NoneType`) – Selector specifying which fields to include in a partial response. Must be a list of fields. For example to get a partial response with just the next page token and the language of each bucket returned: ‘items/id,nextPageToken’

Return type iterable of `gcloud.storage.bucket.Bucket` objects.

Returns All buckets belonging to this project.

lookup_bucket (*bucket_name*)

Get a bucket by name, returning `None` if not found.

You can use this if you would rather check for a `None` value than catching an exception:

```
>>> bucket = client.lookup_bucket('doesn't-exist')
>>> print bucket
None
>>> bucket = client.lookup_bucket('my-bucket')
>>> print bucket
<Bucket: my-bucket>
```

Parameters **bucket_name** (*string*) – The name of the bucket to get.

Return type `gcloud.storage.bucket.Bucket`

Returns The bucket matching the name provided or `None` if not found.

14.1 Connection

Create / interact with gcloud storage connections.

class `gcloud.storage.connection.Connection` (*credentials=None, http=None*)

Bases: `gcloud.connection.JSONConnection`

A connection to Google Cloud Storage via the JSON REST API.

Parameters

- **credentials** (`oauth2client.client.OAuth2Credentials`) – (Optional) The OAuth2 Credentials to use for this connection.
- **http** (`httplib2.Http` or class that defines `request()`) – (Optional) HTTP object to make requests.

API_BASE_URL = ‘https://www.googleapis.com’

The base of the API call URL.

API_URL_TEMPLATE = ‘{api_base_url}/storage/{api_version}/{path}’

A template for the URL of a particular API call.

API_VERSION = 'v1'

The version of the API, used in building the API call's URL.

SCOPE = ('https://www.googleapis.com/auth/devstorage.full_control', 'https://www.googleapis.com/auth/devstorage.read_

The scopes required for authenticating as a Cloud Storage consumer.

Blobs / Objects

Create / interact with Google Cloud Storage blobs.

class `gcloud.storage.blob.Blob` (*name*, *bucket*, *chunk_size=None*)
 Bases: `gcloud.storage._helpers._PropertyMixin`

A wrapper around Cloud Storage's concept of an Object.

Parameters

- **name** (*string*) – The name of the blob. This corresponds to the unique path of the object in the bucket.
- **bucket** (`gcloud.storage.bucket.Bucket`) – The bucket to which this blob belongs.
- **chunk_size** (*integer*) – The size of a chunk of data whenever iterating (1 MB). This must be a multiple of 256 KB per the API specification.

acl

Create our ACL on demand.

cache_control

HTTP 'Cache-Control' header for this object.

See: <https://tools.ietf.org/html/rfc7234#section-5.2> and https://cloud.google.com/storage/docs/json_api/v1/objects

If the property is not set locally, returns `None`.

Return type `string` or `NoneType`

chunk_size

Get the blob's default chunk size.

Return type `integer` or `NoneType`

Returns The current blob's chunk size, if it is set.

client

The client bound to this blob.

component_count

Number of underlying components that make up this object.

See: https://cloud.google.com/storage/docs/json_api/v1/objects

Return type `integer` or `NoneType`

Returns The component count (in case of a composed object) or `None` if the property is not set locally. This property will not be set on objects not created via `compose`.

content_disposition

HTTP 'Content-Disposition' header for this object.

See: <https://tools.ietf.org/html/rfc6266> and https://cloud.google.com/storage/docs/json_api/v1/objects

If the property is not set locally, returns None.

Return type string or NoneType

content_encoding

HTTP 'Content-Encoding' header for this object.

See: <https://tools.ietf.org/html/rfc7231#section-3.1.2.2> and https://cloud.google.com/storage/docs/json_api/v1/objects

If the property is not set locally, returns None.

Return type string or NoneType

content_language

HTTP 'Content-Language' header for this object.

See: <http://tools.ietf.org/html/bcp47> and https://cloud.google.com/storage/docs/json_api/v1/objects

If the property is not set locally, returns None.

Return type string or NoneType

content_type

HTTP 'Content-Type' header for this object.

See: <https://tools.ietf.org/html/rfc2616#section-14.17> and https://cloud.google.com/storage/docs/json_api/v1/objects

If the property is not set locally, returns None.

Return type string or NoneType

crc32c

CRC32C checksum for this object.

See: <http://tools.ietf.org/html/rfc4960#appendix-B> and https://cloud.google.com/storage/docs/json_api/v1/objects

If the property is not set locally, returns None.

Return type string or NoneType

delete (*client=None*)

Deletes a blob from Cloud Storage.

Parameters **client** (*gcloud.storage.client.Client* or NoneType) – Optional.
The client to use. If not passed, falls back to the `client` stored on the blob's bucket.

Return type *Blob*

Returns The blob that was just deleted.

Raises *gcloud.exceptions.NotFound* (propagated from *gcloud.storage.bucket.Bucket.delete_blob()*).

download_as_string (*client=None*)

Download the contents of this blob as a string.

Parameters **client** (*gcloud.storage.client.Client* or NoneType) – Optional.
The client to use. If not passed, falls back to the `client` stored on the blob's bucket.

Return type bytes

Returns The data stored in this blob.

Raises `gcloud.exceptions.NotFound`

download_to_file (*file_obj*, *client=None*)

Download the contents of this blob into a file-like object.

Parameters

- **file_obj** (*file*) – A file handle to which to write the blob’s data.
- **client** (`gcloud.storage.client.Client` or `NoneType`) – Optional. The client to use. If not passed, falls back to the `client` stored on the blob’s bucket.

Raises `gcloud.exceptions.NotFound`

download_to_filename (*filename*, *client=None*)

Download the contents of this blob into a named file.

Parameters

- **filename** (*string*) – A filename to be passed to open.
- **client** (`gcloud.storage.client.Client` or `NoneType`) – Optional. The client to use. If not passed, falls back to the `client` stored on the blob’s bucket.

Raises `gcloud.exceptions.NotFound`

etag

Retrieve the ETag for the object.

See: <http://tools.ietf.org/html/rfc2616#section-3.11> and https://cloud.google.com/storage/docs/json_api/v1/objects

Return type string or `NoneType`

Returns The blob etag or `None` if the property is not set locally.

exists (*client=None*)

Determines whether or not this blob exists.

Parameters **client** (`gcloud.storage.client.Client` or `NoneType`) – Optional. The client to use. If not passed, falls back to the `client` stored on the blob’s bucket.

Return type boolean

Returns True if the blob exists in Cloud Storage.

generate_signed_url (*expiration*, *method='GET'*, *content_type=None*, *generation=None*, *response_disposition=None*, *response_type=None*, *client=None*, *credentials=None*)

Generates a signed URL for this blob.

Note: If you are on Google Compute Engine, you can’t generate a signed URL. Follow [Issue 922](#) for updates on this. If you’d like to be able to generate a signed URL from GCE, you can use a standard service account from a JSON file rather than a GCE service account.

If you have a blob that you want to allow access to for a set amount of time, you can use this method to generate a URL that is only valid within a certain time period.

This is particularly useful if you don’t want publicly accessible blobs, but don’t want to require users to explicitly log in.

Parameters

- **expiration** (*int*, *long*, *datetime.datetime*, *datetime.timedelta*) – When the signed URL should expire.
- **method** (*str*) – The HTTP verb that will be used when requesting the URL.
- **content_type** (*str*) – (Optional) The content type of the object referenced by `resource`.
- **generation** (*str*) – (Optional) A value that indicates which generation of the resource to fetch.
- **response_disposition** (*str*) – (Optional) Content disposition of responses to requests for the signed URL. For example, to enable the signed URL to initiate a file of `blog.png`, use the value `'attachment; filename=blog.png'`.
- **response_type** (*str*) – (Optional) Content type of responses to requests for the signed URL. Used to over-ride the content type of the underlying blob/object.
- **client** (*gcloud.storage.client.Client* or *NoneType*) – (Optional) The client to use. If not passed, falls back to the `client` stored on the blob's bucket.
- **credentials** (*oauth2client.client.OAuth2Credentials* or *NoneType*) – (Optional) The OAuth2 credentials to use to sign the URL. Defaults to the credentials stored on the client used.

Return type `str`

Returns A signed URL you can use to access the resource until expiration.

generation

Retrieve the generation for the object.

See: https://cloud.google.com/storage/docs/json_api/v1/objects

Return type `integer` or `NoneType`

Returns The generation of the blob or `None` if the property is not set locally.

id

Retrieve the ID for the object.

See: https://cloud.google.com/storage/docs/json_api/v1/objects

Return type `string` or `NoneType`

Returns The ID of the blob or `None` if the property is not set locally.

make_public (*client=None*)

Make this blob public giving all users read access.

Parameters **client** (*gcloud.storage.client.Client* or *NoneType*) – Optional. The client to use. If not passed, falls back to the `client` stored on the blob's bucket.

md5_hash

MD5 hash for this object.

See: <http://tools.ietf.org/html/rfc4960#appendix-B> and https://cloud.google.com/storage/docs/json_api/v1/objects

If the property is not set locally, returns `None`.

Return type `string` or `NoneType`

media_link

Retrieve the media download URI for the object.

See: https://cloud.google.com/storage/docs/json_api/v1/objects

Return type string or NoneType

Returns The media link for the blob or None if the property is not set locally.

metadata

Retrieve arbitrary/application specific metadata for the object.

See: https://cloud.google.com/storage/docs/json_api/v1/objects

Return type dict or NoneType

Returns The metadata associated with the blob or None if the property is not set locally.

metageneration

Retrieve the metageneration for the object.

See: https://cloud.google.com/storage/docs/json_api/v1/objects

Return type integer or NoneType

Returns The metageneration of the blob or None if the property is not set locally.

owner

Retrieve info about the owner of the object.

See: https://cloud.google.com/storage/docs/json_api/v1/objects

Return type dict or NoneType

Returns Mapping of owner's role/ID. If the property is not set locally, returns None.

path

Getter property for the URL path to this Blob.

Return type string

Returns The URL path to this Blob.

static path_helper (*bucket_path*, *blob_name*)

Relative URL path for a blob.

Parameters

- **bucket_path** (*string*) – The URL path for a bucket.
- **blob_name** (*string*) – The name of the blob.

Return type string

Returns The relative URL path for blob_name.

public_url

The public URL for this blob's object.

Return type string

Returns The public URL for this blob.

self_link

Retrieve the URI for the object.

See: https://cloud.google.com/storage/docs/json_api/v1/objects

Return type string or NoneType

Returns The self link for the blob or None if the property is not set locally.

size

Size of the object, in bytes.

See: https://cloud.google.com/storage/docs/json_api/v1/objects

Return type integer or `NoneType`

Returns The size of the blob or `None` if the property is not set locally.

storage_class

Retrieve the storage class for the object.

See: <https://cloud.google.com/storage/docs/storage-classes> <https://cloud.google.com/storage/docs/nearline-storage> <https://cloud.google.com/storage/docs/durable-reduced-availability>

Return type string or `NoneType`

Returns If set, one of “STANDARD”, “NEARLINE”, or “DURABLE_REDUCED_AVAILABILITY”, else `None`.

time_deleted

Retrieve the timestamp at which the object was deleted.

See: https://cloud.google.com/storage/docs/json_api/v1/objects

Return type `datetime.datetime` or `NoneType`

Returns Datetime object parsed from RFC3339 valid timestamp, or `None` if the property is not set locally. If the blob has not been deleted, this will never be set.

updated

Retrieve the timestamp at which the object was updated.

See: https://cloud.google.com/storage/docs/json_api/v1/objects

Return type `datetime.datetime` or `NoneType`

Returns Datetime object parsed from RFC3339 valid timestamp, or `None` if the property is not set locally.

upload_from_file (*file_obj*, *rewind=False*, *size=None*, *content_type=None*, *num_retries=6*, *client=None*)

Upload the contents of this blob from a file-like object.

The content type of the upload will either be - The value passed in to the function (if any) - The value stored on the current blob - The default value of ‘application/octet-stream’

Note: The effect of uploading to an existing blob depends on the “versioning” and “lifecycle” policies defined on the blob’s bucket. In the absence of those policies, upload will overwrite any existing contents.

See the [object versioning](#) and [lifecycle](#) API documents for details.

Parameters

- **file_obj** (*file*) – A file handle open for reading.
- **rewind** (*boolean*) – If True, seek to the beginning of the file handle before writing the file to Cloud Storage.
- **size** (*int*) – The number of bytes to read from the file handle. If not provided, we’ll try to guess the size using `os.fstat()`. (If the file handle is not from the filesystem this won’t be possible.)
- **content_type** (string or `NoneType`) – Optional type of content being uploaded.

- **num_retries** (*integer*) – Number of upload retries. Defaults to 6.
- **client** (*gcloud.storage.client.Client* or *NoneType*) – Optional. The client to use. If not passed, falls back to the `client` stored on the blob's bucket.

Raises `ValueError` if size is not passed in and can not be determined

upload_from_filename (*filename, content_type=None, client=None*)

Upload this blob's contents from the content of a named file.

The content type of the upload will either be - The value passed in to the function (if any) - The value stored on the current blob - The value given by `mimetypes.guess_type`

Note: The effect of uploading to an existing blob depends on the “versioning” and “lifecycle” policies defined on the blob's bucket. In the absence of those policies, upload will overwrite any existing contents.

See the [object versioning](#) and [lifecycle](#) API documents for details.

Parameters

- **filename** (*string*) – The path to the file.
- **content_type** (*string* or *NoneType*) – Optional type of content being uploaded.
- **client** (*gcloud.storage.client.Client* or *NoneType*) – Optional. The client to use. If not passed, falls back to the `client` stored on the blob's bucket.

upload_from_string (*data, content_type='text/plain', client=None*)

Upload contents of this blob from the provided string.

Note: The effect of uploading to an existing blob depends on the “versioning” and “lifecycle” policies defined on the blob's bucket. In the absence of those policies, upload will overwrite any existing contents.

See the [object versioning](#) and [lifecycle](#) API documents for details.

Parameters

- **data** (*bytes* or *text*) – The data to store in this blob. If the value is text, it will be encoded as UTF-8.
- **content_type** (*string*) – Optional type of content being uploaded. Defaults to `'text/plain'`.
- **client** (*gcloud.storage.client.Client* or *NoneType*) – Optional. The client to use. If not passed, falls back to the `client` stored on the blob's bucket.

Buckets

Create / interact with gcloud storage buckets.

class `gcloud.storage.bucket.Bucket` (*client, name=None*)
 Bases: `gcloud.storage._helpers._PropertyMixin`

A class representing a Bucket on Cloud Storage.

Parameters

- **client** (*gcloud.storage.client.Client*) – A client which holds credentials and project configuration for the bucket (which requires a project).
- **name** (*string*) – The name of the bucket.

acl

Create our ACL on demand.

blob (*blob_name, chunk_size=None*)
 Factory constructor for blob object.

Note: This will not make an HTTP request; it simply instantiates a blob object owned by this bucket.

Parameters

- **blob_name** (*string*) – The name of the blob to be instantiated.
- **chunk_size** (*integer*) – The size of a chunk of data whenever iterating (1 MB). This must be a multiple of 256 KB per the API specification.

Return type *gcloud.storage.blob.Blob*

Returns The blob object created.

client

The client bound to this bucket.

configure_website (*main_page_suffix=None, not_found_page=None*)
 Configure website-related properties.

See: <https://developers.google.com/storage/docs/website-configuration>

Note: This (apparently) only works if your bucket name is a domain name (and to do that, you need to get approved somehow...).

If you want this bucket to host a website, just provide the name of an index page and a page to use when a blob isn't found:

```
>>> from gcloud import storage
>>> client = storage.Client()
>>> bucket = client.get_bucket(bucket_name)
>>> bucket.configure_website('index.html', '404.html')
```

You probably should also make the whole bucket public:

```
>>> bucket.make_public(recursive=True, future=True)
```

This says: “Make the bucket public, and all the stuff already in the bucket, and anything else I add to the bucket. Just make it all public.”

Parameters

- **main_page_suffix** (*string*) – The page to use as the main page of a directory. Typically something like `index.html`.
- **not_found_page** (*string*) – The file to use when a page isn't found.

copy_blob (*blob, destination_bucket, new_name=None, client=None*)

Copy the given blob to the given bucket, optionally with a new name.

Parameters

- **blob** (*gcloud.storage.blob.Blob*) – The blob to be copied.
- **destination_bucket** (*gcloud.storage.bucket.Bucket*) – The bucket into which the blob should be copied.
- **new_name** (*string*) – (optional) the new name for the copied file.
- **client** (*gcloud.storage.client.Client* or *NoneType*) – Optional. The client to use. If not passed, falls back to the `client` stored on the current bucket.

Return type *gcloud.storage.blob.Blob*

Returns The new Blob.

cors

Retrieve CORS policies configured for this bucket.

See: <http://www.w3.org/TR/cors/> and https://cloud.google.com/storage/docs/json_api/v1/buckets

Return type list of dictionaries

Returns A sequence of mappings describing each CORS policy.

create (*client=None*)

Creates current bucket.

If the bucket already exists, will raise *gcloud.exceptions.Conflict*.

This implements “storage.buckets.insert”.

Parameters **client** (*gcloud.storage.client.Client* or *NoneType*) – Optional. The client to use. If not passed, falls back to the `client` stored on the current bucket.

Return type *gcloud.storage.bucket.Bucket*

Returns The newly created bucket.

default_object_acl

Create our defaultObjectACL on demand.

delete (*force=False, client=None*)

Delete this bucket.

The bucket **must** be empty in order to submit a delete request. If *force=True* is passed, this will first attempt to delete all the objects / blobs in the bucket (i.e. try to empty the bucket).

If the bucket doesn't exist, this will raise `gcloud.exceptions.NotFound`. If the bucket is not empty (and *force=False*), will raise `gcloud.exceptions.Conflict`.

If *force=True* and the bucket contains more than 256 objects / blobs this will cowardly refuse to delete the objects (or the bucket). This is to prevent accidental bucket deletion and to prevent extremely long runtime of this method.

Parameters

- **force** (*boolean*) – If True, empties the bucket's objects then deletes it.
- **client** (`gcloud.storage.client.Client` or `NoneType`) – Optional. The client to use. If not passed, falls back to the `client` stored on the current bucket.

Raises `ValueError` if *force* is True and the bucket contains more than 256 objects / blobs.

delete_blob (*blob_name, client=None*)

Deletes a blob from the current bucket.

If the blob isn't found (backend 404), raises a `gcloud.exceptions.NotFound`.

For example:

```
>>> from gcloud.exceptions import NotFound
>>> from gcloud import storage
>>> client = storage.Client()
>>> bucket = client.get_bucket('my-bucket')
>>> print bucket.list_blobs()
[<Blob: my-bucket, my-file.txt>]
>>> bucket.delete_blob('my-file.txt')
>>> try:
...     bucket.delete_blob('doesnt-exist')
... except NotFound:
...     pass
```

Parameters

- **blob_name** (*string*) – A blob name to delete.
- **client** (`gcloud.storage.client.Client` or `NoneType`) – Optional. The client to use. If not passed, falls back to the `client` stored on the current bucket.

Raises `gcloud.exceptions.NotFound` (to suppress the exception, call `delete_blobs`, passing a no-op `on_error` callback, e.g.:

```
>>> bucket.delete_blobs([blob], on_error=lambda blob: None)
```

delete_blobs (*blobs, on_error=None, client=None*)

Deletes a list of blobs from the current bucket.

Uses `Bucket.delete_blob()` to delete each individual blob.

Parameters

- **blobs** (list of string or `gcloud.storage.blob.Blob`) – A list of blob names or Blob objects to delete.
- **on_error** (a callable taking (blob)) – If not None, called once for each blob raising `gcloud.exceptions.NotFound`; otherwise, the exception is propagated.
- **client** (`gcloud.storage.client.Client` or `NoneType`) – Optional. The client to use. If not passed, falls back to the `client` stored on the current bucket.

Raises `gcloud.exceptions.NotFound` (if `on_error` is not passed).

disable_logging()

Disable access logging for this bucket.

See: <https://cloud.google.com/storage/docs/accesslogs#disabling>

disable_website()

Disable the website configuration for this bucket.

This is really just a shortcut for setting the website-related attributes to None.

enable_logging(bucket_name, object_prefix='')

Enable access logging for this bucket.

See: <https://cloud.google.com/storage/docs/accesslogs#delivery>

Parameters

- **bucket_name** (`string`) – name of bucket in which to store access logs
- **object_prefix** (`string`) – prefix for access log filenames

etag

Retrieve the ETag for the bucket.

See: <http://tools.ietf.org/html/rfc2616#section-3.11> and https://cloud.google.com/storage/docs/json_api/v1/buckets

Return type string or `NoneType`

Returns The bucket etag or None if the property is not set locally.

exists(client=None)

Determines whether or not this bucket exists.

Parameters **client** (`gcloud.storage.client.Client` or `NoneType`) – Optional.

The client to use. If not passed, falls back to the `client` stored on the current bucket.

Return type boolean

Returns True if the bucket exists in Cloud Storage.

get_blob(blob_name, client=None)

Get a blob object by name.

This will return None if the blob doesn't exist:

```
>>> from gcloud import storage
>>> client = storage.Client()
>>> bucket = client.get_bucket('my-bucket')
>>> print bucket.get_blob('/path/to/blob.txt')
<Blob: my-bucket, /path/to/blob.txt>
>>> print bucket.get_blob('/does-not-exist.txt')
None
```


Parameters

- **blob_name** (*string*) – The name of the blob to retrieve.
- **client** (*gcloud.storage.client.Client* or *NoneType*) – Optional. The client to use. If not passed, falls back to the `client` stored on the current bucket.

Return type *gcloud.storage.blob.Blob* or *None*

Returns The blob object if it exists, otherwise *None*.

get_logging()

Return info about access logging for this bucket.

See: <https://cloud.google.com/storage/docs/accesslogs#status>

Return type *dict* or *None*

Returns a *dict* w/ keys, `logBucket` and `logObjectPrefix` (if logging is enabled), or *None* (if not).

id

Retrieve the ID for the bucket.

See: https://cloud.google.com/storage/docs/json_api/v1/buckets

Return type *string* or *NoneType*

Returns The ID of the bucket or *None* if the property is not set locally.

lifecycle_rules

Lifecycle rules configured for this bucket.

See: <https://cloud.google.com/storage/docs/lifecycle> and https://cloud.google.com/storage/docs/json_api/v1/buckets

Return type *list(dict)*

Returns A sequence of mappings describing each lifecycle rule.

list_blobs (*max_results=None, page_token=None, prefix=None, delimiter=None, versions=None, projection='noAcl', fields=None, client=None*)

Return an iterator used to find blobs in the bucket.

Parameters

- **max_results** (*integer* or *NoneType*) – maximum number of blobs to return.
- **page_token** (*string*) – opaque marker for the next “page” of blobs. If not passed, will return the first page of blobs.
- **prefix** (*string* or *NoneType*) – optional prefix used to filter blobs.
- **delimiter** (*string* or *NoneType*) – optional delimiter, used with `prefix` to emulate hierarchy.
- **versions** (*boolean* or *NoneType*) – whether object versions should be returned as separate blobs.
- **projection** (*string* or *NoneType*) – If used, must be ‘full’ or ‘noAcl’. Defaults to ‘noAcl’. Specifies the set of properties to return.
- **fields** (*string* or *NoneType*) – Selector specifying which fields to include in a partial response. Must be a list of fields. For example to get a partial response with just the next page token and the language of each blob returned: ‘items/contentLanguage,nextPageToken’

- **client** (*gcloud.storage.client.Client* or *NoneType*) – Optional. The client to use. If not passed, falls back to the `client` stored on the current bucket.

Return type `_BlobIterator`.

Returns An iterator of blobs.

location

Retrieve location configured for this bucket.

See: https://cloud.google.com/storage/docs/json_api/v1/buckets and <https://cloud.google.com/storage/docs/concepts-techniques#specifyinglocations>

If the property is not set locally, returns `None`.

Return type `string` or `NoneType`

make_public (*recursive=False, future=False, client=None*)

Make a bucket public.

If `recursive=True` and the bucket contains more than 256 objects / blobs this will cowardly refuse to make the objects public. This is to prevent extremely long runtime of this method.

Parameters

- **recursive** (*boolean*) – If `True`, this will make all blobs inside the bucket public as well.
- **future** (*boolean*) – If `True`, this will make all objects created in the future public as well.
- **client** (*gcloud.storage.client.Client* or *NoneType*) – Optional. The client to use. If not passed, falls back to the `client` stored on the current bucket.

metageneration

Retrieve the metageneration for the bucket.

See: https://cloud.google.com/storage/docs/json_api/v1/buckets

Return type `integer` or `NoneType`

Returns The metageneration of the bucket or `None` if the property is not set locally.

owner

Retrieve info about the owner of the bucket.

See: https://cloud.google.com/storage/docs/json_api/v1/buckets

Return type `dict` or `NoneType`

Returns Mapping of owner's role/ID. If the property is not set locally, returns `None`.

path

The URL path to this bucket.

static path_helper (*bucket_name*)

Relative URL path for a bucket.

Parameters **bucket_name** (*string*) – The bucket name in the path.

Return type `string`

Returns The relative URL path for `bucket_name`.

project_number

Retrieve the number of the project to which the bucket is assigned.

See: https://cloud.google.com/storage/docs/json_api/v1/buckets

Return type integer or `NoneType`

Returns The project number that owns the bucket or `None` if the property is not set locally.

rename_blob (*blob*, *new_name*, *client=None*)

Rename the given blob using copy and delete operations.

Effectively, copies blob to the same bucket with a new name, then deletes the blob.

Warning: This method will first duplicate the data and then delete the old blob. This means that with very large objects renaming could be a very (temporarily) costly or a very slow operation.

Parameters

- **blob** (*gcloud.storage.blob.Blob*) – The blob to be renamed.
- **new_name** (*string*) – The new name for this blob.
- **client** (*gcloud.storage.client.Client* or `NoneType`) – Optional. The client to use. If not passed, falls back to the `client` stored on the current bucket.

Return type `Blob`

Returns The newly-renamed blob.

self_link

Retrieve the URI for the bucket.

See: https://cloud.google.com/storage/docs/json_api/v1/buckets

Return type string or `NoneType`

Returns The self link for the bucket or `None` if the property is not set locally.

storage_class

Retrieve the storage class for the bucket.

See: <https://cloud.google.com/storage/docs/storage-classes> <https://cloud.google.com/storage/docs/nearline-storage> <https://cloud.google.com/storage/docs/durable-reduced-availability>

Return type string or `NoneType`

Returns If set, one of “STANDARD”, “NEARLINE”, or “DURABLE_REDUCED_AVAILABILITY”, else `None`.

time_created

Retrieve the timestamp at which the bucket was created.

See: https://cloud.google.com/storage/docs/json_api/v1/buckets

Return type `datetime.datetime` or `NoneType`

Returns Datetime object parsed from RFC3339 valid timestamp, or `None` if the property is not set locally.

versioning_enabled

Is versioning enabled for this bucket?

See: <https://cloud.google.com/storage/docs/object-versioning> for details.

Return type boolean

Returns True if enabled, else False.

ACL

Manipulate access control lists that Cloud Storage provides.

`gcloud.storage.bucket.Bucket` has a getting method that creates an ACL object under the hood, and you can interact with that using `gcloud.storage.bucket.Bucket.acl()`:

```
>>> from gcloud import storage
>>> client = storage.Client()
>>> bucket = client.get_bucket(bucket_name)
>>> acl = bucket.acl
```

Adding and removing permissions can be done with the following methods (in increasing order of granularity):

- `ACL.all()` corresponds to access for all users.
- `ACL.all_authenticated()` corresponds to access for all users that are signed into a Google account.
- `ACL.domain()` corresponds to access on a per Google Apps domain (ie, `example.com`).
- `ACL.group()` corresponds to access on a per group basis (either by ID or e-mail address).
- `ACL.user()` corresponds to access on a per user basis (either by ID or e-mail address).

And you are able to grant and revoke the following roles:

- **Reading:** `_ACLEntity.grant_read()` and `_ACLEntity.revoke_read()`
- **Writing:** `_ACLEntity.grant_write()` and `_ACLEntity.revoke_write()`
- **Owning:** `_ACLEntity.grant_owner()` and `_ACLEntity.revoke_owner()`

You can use any of these like any other factory method (these happen to be `_ACLEntity` factories):

```
>>> acl.user('me@example.org').grant_read()
>>> acl.all_authenticated().grant_write()
```

You can also chain these `grant_*` and `revoke_*` methods together for brevity:

```
>>> acl.all().grant_read().revoke_write()
```

After that, you can save any changes you make with the `gcloud.storage.acl.ACL.save()` method:

```
>>> acl.save()
```

You can alternatively save any existing `gcloud.storage.acl.ACL` object (whether it was created by a factory method or not) from a `gcloud.storage.bucket.Bucket`:

```
>>> bucket.acl.save(acl=acl)
```

To get the list of `entity` and `role` for each unique pair, the `ACL` class is iterable:

```
>>> print list(ACL)
[{'role': 'OWNER', 'entity': 'allUsers'}, ...]
```

This list of tuples can be used as the `entity` and `role` fields when sending metadata for ACLs to the API.

class `gcloud.storage.acl.ACL`

Bases: `object`

Container class representing a list of access controls.

add_entity (*entity*)

Add an entity to the ACL.

Parameters `entity` (`_ACLEntity`) – The entity to add to this ACL.

all ()

Factory method for an Entity representing all users.

Return type `_ACLEntity`

Returns An entity representing all users.

all_authenticated ()

Factory method for an Entity representing all authenticated users.

Return type `_ACLEntity`

Returns An entity representing all authenticated users.

clear (*client=None*)

Remove all ACL entries.

Note that this won't actually remove *ALL* the rules, but it will remove all the non-default rules. In short, you'll still have access to a bucket that you created even after you clear ACL rules with this method.

Parameters `client` (`gcloud.storage.client.Client` or `NoneType`) – Optional. The client to use. If not passed, falls back to the `client` stored on the ACL's parent.

client

Abstract getter for the object client.

domain (*domain*)

Factory method for a domain Entity.

Parameters `domain` (*string*) – The domain for this entity.

Return type `_ACLEntity`

Returns An entity corresponding to this domain.

entity (*entity_type, identifier=None*)

Factory method for creating an Entity.

If an entity with the same type and identifier already exists, this will return a reference to that entity. If not, it will create a new one and add it to the list of known entities for this ACL.

Parameters

- **entity_type** (*string*) – The type of entity to create (ie, `user`, `group`, etc)
- **identifier** (*string*) – The ID of the entity (if applicable). This can be either an ID or an e-mail address.

Return type `_ACLEntity`

Returns A new Entity or a reference to an existing identical entity.

entity_from_dict (*entity_dict*)

Build an `_ACLEntity` object from a dictionary of data.

An entity is a mutable object that represents a list of roles belonging to either a user or group or the special types for all users and all authenticated users.

Parameters **entity_dict** (*dict*) – Dictionary full of data from an ACL lookup.

Return type `_ACLEntity`

Returns An Entity constructed from the dictionary.

get_entities ()

Get a list of all Entity objects.

Return type list of `_ACLEntity` objects

Returns A list of all Entity objects.

get_entity (*entity*, *default=None*)

Gets an entity object from the ACL.

Parameters

- **entity** (`_ACLEntity` or string) – The entity to get lookup in the ACL.
- **default** (*anything*) – This value will be returned if the entity doesn't exist.

Return type `_ACLEntity`

Returns The corresponding entity or the value provided to `default`.

group (*identifier*)

Factory method for a group Entity.

Parameters **identifier** (*string*) – An id or e-mail for this particular group.

Return type `_ACLEntity`

Returns An Entity corresponding to this group.

has_entity (*entity*)

Returns whether or not this ACL has any entries for an entity.

Parameters **entity** (`_ACLEntity`) – The entity to check for existence in this ACL.

Return type boolean

Returns True if the entity exists in the ACL.

loaded = False

reload (*client=None*)

Reload the ACL data from Cloud Storage.

Parameters **client** (`gcloud.storage.client.Client` or `NoneType`) – Optional.
The client to use. If not passed, falls back to the `client` stored on the ACL's parent.

reload_path = None

reset ()

Remove all entities from the ACL, and clear the `loaded` flag.

save (*acl=None*, *client=None*)

Save this ACL for the current bucket.

Parameters

- **acl** (*gcloud.storage.acl.ACL*, or a compatible list.) – The ACL object to save. If left blank, this will save current entries.
- **client** (*gcloud.storage.client.Client* or `NoneType`) – Optional. The client to use. If not passed, falls back to the `client` stored on the ACL's parent.

save_path = `None`**save_predefined** (*predefined, client=None*)

Save this ACL for the current bucket using a predefined ACL.

Parameters

- **predefined** (*string*) – An identifier for a predefined ACL. Must be one of the keys in `_PREDEFINED_ACLS`. If passed, `acl` must be `None`.
- **client** (*gcloud.storage.client.Client* or `NoneType`) – Optional. The client to use. If not passed, falls back to the `client` stored on the ACL's parent.

user (*identifier*)

Factory method for a user Entity.

Parameters **identifier** (*string*) – An id or e-mail for this particular user.**Return type** `_ACLEntity`**Returns** An Entity corresponding to this user.**class** `gcloud.storage.acl.BucketACL` (*bucket*)Bases: *gcloud.storage.acl.ACL*

An ACL specifically for a bucket.

Parameters **bucket** (*gcloud.storage.bucket.Bucket*) – The bucket to which this ACL relates.**client**

The client bound to this ACL's bucket.

reload_path

Compute the path for GET API requests for this ACL.

save_path

Compute the path for PATCH API requests for this ACL.

class `gcloud.storage.acl.DefaultObjectACL` (*bucket*)Bases: *gcloud.storage.acl.BucketACL*

A class representing the default object ACL for a bucket.

class `gcloud.storage.acl.ObjectACL` (*blob*)Bases: *gcloud.storage.acl.ACL*

An ACL specifically for a Cloud Storage object / blob.

Parameters **blob** (*gcloud.storage.blob.Blob*) – The blob that this ACL corresponds to.**client**

The client bound to this ACL's blob.

reload_path

Compute the path for GET API requests for this ACL.

save_path

Compute the path for PATCH API requests for this ACL.

Batches

Batch updates / deletes of storage buckets / blobs.

See: https://cloud.google.com/storage/docs/json_api/v1/how-tos/batch

class `gcloud.storage.batch.Batch` (*client*)

Bases: `gcloud.storage.connection.Connection`

Proxy an underlying connection, batching up change operations.

Parameters `client` (`gcloud.storage.client.Client`) – The client to use for making connections.

current ()

Return the topmost batch, or None.

finish ()

Submit a single *multipart/mixed* request w/ deferred requests.

Return type list of tuples

Returns one (headers, payload) tuple per deferred request.

class `gcloud.storage.batch.MIMEApplicationHTTP` (*method, uri, headers, body*)

Bases: `email.mime.application.MIMEApplication`

MIME type for application/http.

Constructs payload from headers and body

Parameters

- **method** (*string*) – HTTP method
- **uri** (*string*) – URI for HTTP request
- **headers** (*dict*) – HTTP headers
- **body** (*text or None*) – HTTP payload

class `gcloud.storage.batch.NoContent`

Bases: `object`

Emulate an HTTP '204 No Content' response.

status = 204

Using the API

19.1 Authentication / Configuration

- Use *Client* objects to configure your applications.
- *Client* objects hold both a `project` and an authenticated connection to the PubSub service.
- The authentication credentials can be implicitly determined from the environment or directly via `from_service_account_json` and `from_service_account_p12`.
- After setting `GOOGLE_APPLICATION_CREDENTIALS` and `GCLOUD_PROJECT` environment variables, create a *Client*

```
>>> from gcloud import pubsub
>>> client = pubsub.Client()
```

19.2 Manage topics for a project

Create a new topic for the default project:

```
>>> from gcloud import pubsub
>>> client = pubsub.Client()
>>> topic = client.topic('topic_name')
>>> topic.create() # API request
```

Check for the existence of a topic:

```
>>> from gcloud import pubsub
>>> client = pubsub.Client()
>>> topic = client.topic('topic_name')
>>> topic.exists() # API request
True
```

List topics for the default project:

```
>>> from gcloud import pubsub
>>> client = pubsub.Client()
>>> topics, next_page_token = client.list_topics() # API request
>>> [topic.name for topic in topics]
['topic_name']
```

Delete a topic:

```
>>> from gcloud import pubsub
>>> client = pubsub.Client()
>>> topic = client.topic('topic_name')
>>> topic.delete() # API request
```

19.3 Publish messages to a topic

Publish a single message to a topic, without attributes:

```
>>> from gcloud import pubsub
>>> client = pubsub.Client()
>>> topic = client.topic('topic_name')
>>> topic.publish('this is the message_payload') # API request
<message_id>
```

Publish a single message to a topic, with attributes:

```
>>> from gcloud import pubsub
>>> client = pubsub.Client()
>>> topic = client.topic('topic_name')
>>> topic.publish('this is another message_payload',
...              attr1='value1', attr2='value2') # API request
<message_id>
```

Publish a set of messages to a topic (as a single request):

```
>>> from gcloud import pubsub
>>> client = pubsub.Client()
>>> topic = client.topic('topic_name')
>>> with topic.batch() as batch:
...     batch.publish('this is the first message_payload')
...     batch.publish('this is the second message_payload',
...                   attr1='value1', attr2='value2')
>>> list(batch)
[<message_id1>, <message_id2>]
```

Note: The only API request happens during the `__exit__()` of the topic used as a context manager.

19.4 Manage subscriptions to topics

Create a new pull subscription for a topic:

```
>>> from gcloud import pubsub
>>> client = pubsub.Client()
>>> topic = client.topic('topic_name')
>>> subscription = topic.subscription('subscription_name')
>>> subscription.create() # API request
```

Create a new pull subscription for a topic with a non-default ACK deadline:

```
>>> from gcloud import pubsub
>>> client = pubsub.Client()
>>> topic = client.topic('topic_name')
```

```
>>> subscription = topic.subscription('subscription_name', ack_deadline=90)
>>> subscription.create() # API request
```

Create a new push subscription for a topic:

```
>>> from gcloud import pubsub
>>> ENDPOINT = 'https://example.com/hook'
>>> client = pubsub.Client()
>>> topic = client.topic('topic_name')
>>> subscription = topic.subscription('subscription_name',
...                               push_endpoint=ENDPOINT)
>>> subscription.create() # API request
```

Check for the existence of a subscription:

```
>>> from gcloud import pubsub
>>> client = pubsub.Client()
>>> topic = client.topic('topic_name')
>>> subscription = topic.subscription('subscription_name')
>>> subscription.exists() # API request
True
```

Convert a pull subscription to push:

```
>>> from gcloud import pubsub
>>> ENDPOINT = 'https://example.com/hook'
>>> client = pubsub.Client()
>>> topic = client.topic('topic_name')
>>> subscription = topic.subscription('subscription_name')
>>> subscription.modify_push_configuration(push_endpoint=ENDPOINT) # API request
```

Convert a push subscription to pull:

```
>>> from gcloud import pubsub
>>> ENDPOINT = 'https://example.com/hook'
>>> client = pubsub.Client()
>>> topic = client.topic('topic_name')
>>> subscription = topic.subscription('subscription_name',
...                               push_endpoint=ENDPOINT)
>>> subscription.modify_push_configuration(push_endpoint=None) # API request
```

List subscriptions for a topic:

```
>>> from gcloud import pubsub
>>> client = pubsub.Client()
>>> subscriptions, next_page_token = client.list_subscriptions(
...     topic_name='topic_name') # API request
>>> [subscription.name for subscription in subscriptions]
['subscription_name']
```

List all subscriptions for the default project:

```
>>> from gcloud import pubsub
>>> client = pubsub.Client()
>>> subscription, next_page_tokens = client.list_subscriptions() # API request
>>> [subscription.name for subscription in subscriptions]
['subscription_name']
```

Delete a subscription:

```
>>> from gcloud import pubsub
>>> client = pubsub.Client()
>>> topic = client.topic('topic_name')
>>> subscription = topic.subscription('subscription_name')
>>> subscription.delete() # API request
```

19.5 Pull messages from a subscription

Fetch pending messages for a pull subscription:

```
>>> from gcloud import pubsub
>>> client = pubsub.Client()
>>> topic = client.topic('topic_name')
>>> subscription = topic.subscription('subscription_name')
>>> with topic.batch() as batch:
...     batch.publish('this is the first message_payload')
...     batch.publish('this is the second message_payload',
...                   attr1='value1', attr2='value2')
>>> received = subscription.pull() # API request
>>> messages = [recv[1] for recv in received]
>>> [message.message_id for message in messages]
[<message_id1>, <message_id2>]
>>> [message.data for message in messages]
['this is the first message_payload', 'this is the second message_payload']
>>> [message.attributes for message in messages]
[{}, {'attr1': 'value1', 'attr2': 'value2'}]
```

Note that received messages must be acknowledged, or else the back-end will re-send them later:

```
>>> ack_ids = [recv[0] for recv in received]
>>> subscription.acknowledge(ack_ids)
```

Fetch a limited number of pending messages for a pull subscription:

```
>>> from gcloud import pubsub
>>> client = pubsub.Client()
>>> topic = client.topic('topic_name')
>>> subscription = topic.subscription('subscription_name')
>>> with topic.batch() as batch:
...     batch.publish('this is the first message_payload')
...     batch.publish('this is the second message_payload',
...                   attr1='value1', attr2='value2')
>>> received = subscription.pull(max_messages=1) # API request
>>> messages = [recv[1] for recv in received]
>>> [message.message_id for message in messages]
```

Fetch messages for a pull subscription without blocking (none pending):

```
>>> from gcloud import pubsub
>>> client = pubsub.Client()
>>> topic = client.topic('topic_name')
>>> subscription = topic.subscription('subscription_name')
>>> received = subscription.pull(return_immediately=True) # API request
>>> messages = [recv[1] for recv in received]
>>> [message.message_id for message in messages]
[]
```

Pub/Sub Client

Client for interacting with the Google Cloud Pub/Sub API.

class `gcloud.pubsub.client.Client` (*project=None, credentials=None, http=None*)
 Bases: `gcloud.client.JSONClient`

Client to bundle configuration needed for API requests.

Parameters

- **project** (*string*) – the project which the client acts on behalf of. Will be passed when creating a topic. If not passed, falls back to the default inferred from the environment.
- **credentials** (`oauth2client.client.OAuth2Credentials` or `NoneType`) – The OAuth2 Credentials to use for the connection owned by this client. If not passed (and if no `http` object is passed), falls back to the default inferred from the environment.
- **http** (`httplib2.Http` or class that defines `request()`) – An optional HTTP object to make requests. If not passed, an `http` object is created that is bound to the `credentials` for the current object.

list_subscriptions (*page_size=None, page_token=None, topic_name=None*)

List subscriptions for the project associated with this client.

See: <https://cloud.google.com/pubsub/reference/rest/v1/projects/topics/list>

and (where `topic_name` is passed): <https://cloud.google.com/pubsub/reference/rest/v1/projects/topics/subscriptions/list>

Parameters

- **page_size** (*int*) – maximum number of topics to return, If not passed, defaults to a value set by the API.
- **page_token** (*string*) – opaque marker for the next “page” of topics. If not passed, the API will return the first page of topics.
- **topic_name** (*string*) – limit results to subscriptions bound to the given topic.

Return type tuple, (list, str)

Returns list of `gcloud.pubsub.subscription.Subscription`, plus a “next page token” string: if not `None`, indicates that more topics can be retrieved with another call (pass that value as `page_token`).

list_topics (*page_size=None, page_token=None*)

List topics for the project associated with this client.

See: <https://cloud.google.com/pubsub/reference/rest/v1/projects/topics/list>

Parameters

- **page_size** (*int*) – maximum number of topics to return, If not passed, defaults to a value set by the API.
- **page_token** (*string*) – opaque marker for the next “page” of topics. If not passed, the API will return the first page of topics.

Return type tuple, (list, str)

Returns list of `gcloud.pubsub.topic.Topic`, plus a “next page token” string: if not None, indicates that more topics can be retrieved with another call (pass that value as `page_token`).

topic (*name*, *timestamp_messages=False*)
Creates a topic bound to the current client.

Parameters

- **name** (*string*) – the name of the topic to be constructed.
- **timestamp_messages** (*boolean*) – To be passed to Topic constructor.

Return type `gcloud.pubsub.topic.Topic`

Returns Topic created with the current client.

20.1 Connection

Create / interact with gcloud pubsub connections.

```
class gcloud.pubsub.connection.Connection (credentials=None, http=None,
                                           api_base_url=None)
```

Bases: `gcloud.connection.JSONConnection`

A connection to Google Cloud Pubsub via the JSON REST API.

Parameters

- **credentials** (`oauth2client.client.OAuth2Credentials`) – (Optional) The OAuth2 Credentials to use for this connection.
- **http** (`httplib2.Http` or class that defines `request()`) – (Optional) HTTP object to make requests.
- **api_base_url** (*string*) – The base of the API call URL. Defaults to the value `Connection.API_BASE_URL`.

API_BASE_URL = ‘https://pubsub.googleapis.com’

The base of the API call URL.

API_URL_TEMPLATE = ‘{api_base_url}/{api_version}{path}’

A template for the URL of a particular API call.

API_VERSION = ‘v1’

The version of the API, used in building the API call’s URL.

SCOPE = (‘https://www.googleapis.com/auth/pubsub’, ‘https://www.googleapis.com/auth/cloud-platform’)

The scopes required for authenticating as a Cloud Pub/Sub consumer.

build_api_url (*path*, *query_params=None*, *api_base_url=None*, *api_version=None*)

Construct an API url given a few components, some optional.

Typically, you shouldn't need to use this method.

Parameters

- **path** (*string*) – The path to the resource.
- **query_params** (*dict*) – A dictionary of keys and values to insert into the query string of the URL.
- **api_base_url** (*string*) – The base URL for the API endpoint. Typically you won't have to provide this.
- **api_version** (*string*) – The version of the API to call. Typically you shouldn't provide this and instead use the default for the library.

Return type *string***Returns** The URL assembled from the pieces provided.

Topics

Define API Topics.

class `gcloud.pubsub.topic.Batch(topic, client)`

Bases: `object`

Context manager: collect messages to publish via a single API call.

Helper returned by `:meth:Topic.batch`

Parameters

- **topic** (`gcloud.pubsub.topic.Topic`) – the topic being published
- **client** (`gcloud.pubsub.client.Client`) – The client to use.

commit (`client=None`)

Send saved messages as a single API call.

Parameters **client** (`gcloud.pubsub.client.Client` or `NoneType`) – the client to use. If not passed, falls back to the `client` stored on the current batch.

publish (`message, **attrs`)

Emulate publishing a message, but save it.

Parameters

- **message** (`bytes`) – the message payload
- **attrs** (`dict (string -> string)`) – key-value pairs to send as message attributes

class `gcloud.pubsub.topic.Topic(name, client, timestamp_messages=False)`

Bases: `object`

Topics are targets to which messages can be published.

Subscribers then receive those messages.

See: <https://cloud.google.com/pubsub/reference/rest/v1/projects/topics>

Parameters

- **name** (`string`) – the name of the topic
- **client** (`gcloud.pubsub.client.Client`) – A client which holds credentials and project configuration for the topic (which requires a project).
- **timestamp_messages** (`boolean`) – If true, the topic will add a `timestamp` key to the attributes of each published message: the value will be an RFC 3339 timestamp.

batch (*client=None*)

Return a batch to use as a context manager.

Parameters **client** (*gcloud.pubsub.client.Client* or `NoneType`) – the client to use. If not passed, falls back to the `client` stored on the current topic.

Return type *Batch*

Returns A batch to use as a context manager.

create (*client=None*)

API call: create the topic via a PUT request

See: <https://cloud.google.com/pubsub/reference/rest/v1/projects/topics/create>

Parameters **client** (*gcloud.pubsub.client.Client* or `NoneType`) – the client to use. If not passed, falls back to the `client` stored on the current topic.

delete (*client=None*)

API call: delete the topic via a DELETE request

See: <https://cloud.google.com/pubsub/reference/rest/v1/projects/topics/delete>

Parameters **client** (*gcloud.pubsub.client.Client* or `NoneType`) – the client to use. If not passed, falls back to the `client` stored on the current topic.

exists (*client=None*)

API call: test for the existence of the topic via a GET request

See <https://cloud.google.com/pubsub/reference/rest/v1/projects/topics/get>

Parameters **client** (*gcloud.pubsub.client.Client* or `NoneType`) – the client to use. If not passed, falls back to the `client` stored on the current topic.

classmethod from_api_repr (*resource, client*)

Factory: construct a topic given its API representation

Parameters

- **resource** (*dict*) – topic resource representation returned from the API
- **client** (*gcloud.pubsub.client.Client*) – Client which holds credentials and project configuration for the topic.

Return type *gcloud.pubsub.topic.Topic*

Returns Topic parsed from `resource`.

Raises `ValueError` if `client` is not `None` and the project from the resource does not agree with the project from the client.

full_name

Fully-qualified name used in topic / subscription APIs

path

URL path for the topic's APIs

project

Project bound to the topic.

publish (*message, client=None, **attrs*)

API call: publish a message to a topic via a POST request

See: <https://cloud.google.com/pubsub/reference/rest/v1/projects/topics/publish>

Parameters

- **message** (*bytes*) – the message payload
- **client** (*gcloud.pubsub.client.Client* or *NoneType*) – the client to use. If not passed, falls back to the `client` stored on the current topic.
- **attrs** (*dict (string -> string)*) – key-value pairs to send as message attributes

Return type *str*

Returns message ID assigned by the server to the published message

subscription (*name, ack_deadline=None, push_endpoint=None*)

Creates a subscription bound to the current topic.

Parameters

- **name** (*string*) – the name of the subscription
- **ack_deadline** (*int*) – the deadline (in seconds) by which messages pulled from the back-end must be acknowledged.
- **push_endpoint** (*string*) – URL to which messages will be pushed by the back-end. If not set, the application must pull messages.

Subscriptions

Define API Subscriptions.

```
class gcloud.pubsub.subscription.Subscription(name, topic, ack_deadline=None,
                                             push_endpoint=None)
```

Bases: `object`

Subscriptions receive messages published to their topics.

See: <https://cloud.google.com/pubsub/reference/rest/v1/projects.subscriptions>

Parameters

- **name** (*string*) – the name of the subscription
- **topic** (*gcloud.pubsub.topic.Topic*) – the topic to which the subscription belongs..
- **ack_deadline** (*int*) – the deadline (in seconds) by which messages pulled from the back-end must be acknowledged.
- **push_endpoint** (*string*) – URL to which messages will be pushed by the back-end. If not set, the application must pull messages.

acknowledge (*ack_ids, client=None*)

API call: acknowledge retrieved messages for the subscription.

See: <https://cloud.google.com/pubsub/reference/rest/v1/projects.subscriptions/acknowledge>

Parameters

- **ack_ids** (*list of string*) – ack IDs of messages being acknowledged
- **client** (*gcloud.pubsub.client.Client* or `NoneType`) – the client to use. If not passed, falls back to the `client` stored on the current subscription's topic.

create (*client=None*)

API call: create the subscription via a PUT request

See: <https://cloud.google.com/pubsub/reference/rest/v1/projects.subscriptions/create>

Parameters **client** (*gcloud.pubsub.client.Client* or `NoneType`) – the client to use. If not passed, falls back to the `client` stored on the current subscription's topic.

delete (*client=None*)

API call: delete the subscription via a DELETE request.

See: <https://cloud.google.com/pubsub/reference/rest/v1/projects.subscriptions/delete>

Parameters `client` (`gcloud.pubsub.client.Client` or `NoneType`) – the client to use. If not passed, falls back to the `client` stored on the current subscription's topic.

exists (`client=None`)

API call: test existence of the subscription via a GET request

See <https://cloud.google.com/pubsub/reference/rest/v1/projects.subscriptions/get>

Parameters `client` (`gcloud.pubsub.client.Client` or `NoneType`) – the client to use. If not passed, falls back to the `client` stored on the current subscription's topic.

classmethod from_api_repr (`resource`, `client`, `topics=None`)

Factory: construct a topic given its API representation

Parameters

- **resource** (`dict`) – topic resource representation returned from the API
- **client** (`gcloud.pubsub.client.Client`) – Client which holds credentials and project configuration for a topic.
- **topics** (`dict` or `None`) – A mapping of topic names -> topics. If not passed, the subscription will have a newly-created topic.

Return type `gcloud.pubsub.subscription.Subscription`

Returns Subscription parsed from `resource`.

modify_ack_deadline (`ack_id`, `ack_deadline`, `client=None`)

API call: update acknowledgement deadline for a retrieved message.

See: <https://cloud.google.com/pubsub/reference/rest/v1/projects.subscriptions/acknowledge>

Parameters

- **ack_id** (`string`) – ack ID of message being updated
- **ack_deadline** (`int`) – new deadline for the message, in seconds
- **client** (`gcloud.pubsub.client.Client` or `NoneType`) – the client to use. If not passed, falls back to the `client` stored on the current subscription's topic.

modify_push_configuration (`push_endpoint`, `client=None`)

API call: update the push endpoint for the subscription.

See: <https://cloud.google.com/pubsub/reference/rest/v1/projects.subscriptions/modifyPushConfig>

Parameters

- **push_endpoint** (`string`) – URL to which messages will be pushed by the back-end. If `None`, the application must pull messages.
- **client** (`gcloud.pubsub.client.Client` or `NoneType`) – the client to use. If not passed, falls back to the `client` stored on the current subscription's topic.

path

URL path for the subscription's APIs

pull (`return_immediately=False`, `max_messages=1`, `client=None`)

API call: retrieve messages for the subscription.

See: <https://cloud.google.com/pubsub/reference/rest/v1/projects.subscriptions/pull>

Parameters

- **return_immediately** (`boolean`) – if `True`, the back-end returns even if no messages are available; if `False`, the API call blocks until one or more messages are available.

- **max_messages** (*int*) – the maximum number of messages to return.
- **client** (*gcloud.pubsub.client.Client* or *NoneType*) – the client to use. If not passed, falls back to the `client` stored on the current subscription's topic.

Return type list of (ack_id, message) tuples

Returns sequence of tuples: `ack_id` is the ID to be used in a subsequent call to `acknowledge()`, and `message` is an instance of *gcloud.pubsub.message.Message*.

reload (*client=None*)

API call: sync local subscription configuration via a GET request

See <https://cloud.google.com/pubsub/reference/rest/v1/projects.subscriptions/get>

Parameters **client** (*gcloud.pubsub.client.Client* or *NoneType*) – the client to use. If not passed, falls back to the `client` stored on the current subscription's topic.

Message

Define API Topics.

class `gcloud.pubsub.message.Message` (*data*, *message_id*, *attributes=None*)

Bases: `object`

Messages can be published to a topic and received by subscribers.

See: <https://cloud.google.com/pubsub/reference/rest/v1/PubsubMessage>

Parameters

- **data** (*bytes*) – the payload of the message
- **message_id** (*string*) – An ID assigned to the message by the API.
- **attributes** (*dict or None*) – Extra metadata associated by the publisher with the message.

attributes

Lazily-constructed attribute dictionary

classmethod `from_api_repr` (*api_repr*)

Factory: construct message from API representation.

Parameters `api_repr` (*dict or None*) – The API representation of the message

timestamp

Return sortable timestamp from attributes, if passed.

Allows sorting messages in publication order (assuming consistent clocks across all publishers).

Return type `datetime.datetime`

Returns timestamp (in UTC timezone) parsed from RFC 3339 timestamp

Raises `ValueError` if timestamp not in `attributes`, or if it does not match the RFC 3339 format.

Using the API

24.1 Authentication / Configuration

- Use *Client* objects to configure your applications.
- *Client* objects hold both a `project` and an authenticated connection to the BigQuery service.
- The authentication credentials can be implicitly determined from the environment or directly via `from_service_account_json` and `from_service_account_p12`.
- After setting `GOOGLE_APPLICATION_CREDENTIALS` and `GLOUD_PROJECT` environment variables, create an instance of *Client*.

```
>>> from gcloud import bigquery
>>> client = bigquery.Client()
```

- Override the credentials inferred from the environment by passing explicit credentials to one of the alternative classmethod factories, `gcloud.bigquery.client.Client.from_service_account_json()`:

```
>>> from gcloud import bigquery
>>> client = bigquery.Client.from_service_account_json('/path/to/creds.json')
```

or `gcloud.bigquery.client.Client.from_service_account_p12()`:

```
>>> from gcloud import bigquery
>>> client = bigquery.Client.from_service_account_p12(
...     '/path/to/creds.p12', 'jrandom@example.com')
```

24.2 Projects

A project is the top-level container in the BigQuery API: it is tied closely to billing, and can provide default access control across all its datasets. If no `project` is passed to the client container, the library attempts to infer a project using the environment (including explicit environment variables, GAE, and GCE).

To override the project inferred from the environment, pass an explicit `project` to the constructor, or to either of the alternative classmethod factories:

```
>>> from gcloud import bigquery
>>> client = bigquery.Client(project='PROJECT_ID')
```

24.2.1 Project ACLs

Each project has an access control list granting reader / writer / owner permission to one or more entities. This list cannot be queried or set via the API: it must be managed using the Google Developer Console.

24.3 Datasets

A dataset represents a collection of tables, and applies several default policies to tables as they are created:

- An access control list (ACL). When created, a dataset has an ACL which maps to the ACL inherited from its project.
- A default table expiration period. If set, tables created within the dataset will have the value as their expiration period.

24.3.1 Dataset operations

Create a new dataset for the client's project:

```
>>> from gcloud import bigquery
>>> client = bigquery.Client()
>>> dataset = client.dataset('dataset_name')
>>> dataset.create() # API request
```

Check for the existence of a dataset:

```
>>> from gcloud import bigquery
>>> client = bigquery.Client()
>>> dataset = client.dataset('dataset_name')
>>> dataset.exists() # API request
True
```

List datasets for the client's project:

```
>>> from gcloud import bigquery
>>> client = bigquery.Client()
>>> datasets, next_page_token = client.list_datasets() # API request
>>> [dataset.name for dataset in datasets]
['dataset_name']
```

Refresh metadata for a dataset (to pick up changes made by another client):

```
>>> from gcloud import bigquery
>>> client = bigquery.Client()
>>> dataset = client.dataset('dataset_name')
>>> dataset.reload() # API request
```

Patch metadata for a dataset:

```
>>> from gcloud import bigquery
>>> client = bigquery.Client()
>>> dataset = client.dataset('dataset_name')
>>> one_day_ms = 24 * 60 * 60 * 1000
>>> dataset.patch(description='Description goes here',
...               default_table_expiration_ms=one_day_ms) # API request
```

Replace the ACL for a dataset, and update all writeable fields:


```
>>> from gcloud import bigquery
>>> client = bigquery.Client()
>>> dataset = client.dataset('dataset_name')
>>> dataset.get() # API request
>>> acl = list(dataset.acl)
>>> acl.append(bigquery.Access(role='READER', entity_type='domain', entity='example.com'))
>>> dataset.acl = acl
>>> dataset.update() # API request
```

Delete a dataset:

```
>>> from gcloud import bigquery
>>> client = bigquery.Client()
>>> dataset = client.dataset('dataset_name')
>>> dataset.delete() # API request
```

24.4 Tables

Tables exist within datasets. List tables for the dataset:

```
>>> from gcloud import bigquery
>>> client = bigquery.Client()
>>> dataset = client.dataset('dataset_name')
>>> tables, next_page_token = dataset.list_tables() # API request
>>> [table.name for table in tables]
['table_name']
```

Create a table:

```
>>> from gcloud import bigquery
>>> client = bigquery.Client()
>>> dataset = client.dataset('dataset_name')
>>> table = dataset.table(name='person_ages')
>>> table.create() # API request
```

Check for the existence of a table:

```
>>> from gcloud import bigquery
>>> client = bigquery.Client()
>>> dataset = client.dataset('dataset_name')
>>> table = dataset.table(name='person_ages')
>>> table.exists() # API request
True
```

Refresh metadata for a table (to pick up changes made by another client):

```
>>> from gcloud import bigquery
>>> client = bigquery.Client()
>>> dataset = client.dataset('dataset_name')
>>> dataset.reload() # API request
```

Patch specific properties for a table:

```
>>> from gcloud import bigquery
>>> client = bigquery.Client()
>>> dataset = client.dataset('dataset_name')
>>> table = dataset.table(name='person_ages')
```

```
>>> table.patch(friendly_name='Person Ages',
...             description='Ages of persons') # API request
```

Update all writable metadata for a table

```
>>> from gcloud import bigquery
>>> from gcloud.bigquery import SchemaField
>>> client = bigquery.Client()
>>> dataset = client.dataset('dataset_name')
>>> table = dataset.table(name='person_ages')
>>> table.schema = [
...     SchemaField(name='full_name', type='string', mode='required'),
...     SchemaField(name='age', type='int', mode='required')]
>>> table.update() # API request
```

Upload table data from a file:

```
>>> from gcloud import bigquery
>>> client = bigquery.Client()
>>> dataset = client.dataset('dataset_name')
>>> table = dataset.table(name='person_ages')
>>> table.schema = [
...     SchemaField(name='full_name', type='string', mode='required'),
...     SchemaField(name='age', type='int', mode='required')]
>>> with open('person_ages.csv', 'rb') as csv_file:
...     table.upload_from_file(csv_file, CSV,
...                             create_disposition='CREATE_IF_NEEDED')
```

Get rows from a table's data:

```
>>> from gcloud import bigquery
>>> client = bigquery.Client()
>>> dataset = client.dataset('dataset_name')
>>> table = dataset.table(name='person_ages')
>>> rows, next_page_token = table.fetch_data(max_results=100) # API request
>>> for row in rows:
...     for field, value in zip(table.schema, row):
...         do_something(field, value)
```

Delete a table:

```
>>> from gcloud import bigquery
>>> client = bigquery.Client()
>>> dataset = client.dataset('dataset_name')
>>> table = dataset.table(name='person_ages')
>>> table.delete() # API request
```

24.5 Jobs

Jobs describe actions performed on data in BigQuery tables:

- Load data into a table
- Run a query against data in one or more tables
- Extract data from a table
- Copy a table

List jobs for a project:

```
>>> from gcloud import bigquery
>>> client = bigquery.Client()
>>> jobs, token = client.list_jobs() # API request
>>> [(job.name, job.job_type, job.created, job.state) for job in jobs]
['load-table-job', 'load', (datetime.datetime(2015, 7, 23, 9, 30, 20, 268260, tzinfo=<UTC>), 'done')]
```

24.5.1 Querying data (synchronous)

Run a query which can be expected to complete within bounded time:

```
>>> from gcloud import bigquery
>>> client = bigquery.Client()
>>> query = """\
SELECT count(*) AS age_count FROM dataset_name.person_ages
"""
>>> query = client.run_sync_query(query)
>>> query.timeout_ms = 1000
>>> query.run() # API request
>>> retry_count = 100
>>> while retry_count > 0 and not job.complete:
...     retry_count -= 1
...     time.sleep(10)
...     query.reload() # API request
>>> query.schema
[{'name': 'age_count', 'type': 'integer', 'mode': 'nullable'}]
>>> query.rows
[(15,)]
```

Note: If the query takes longer than the timeout allowed, `job.complete` will be `False`: we therefore poll until it is completed.

24.5.2 Querying data (asynchronous)

Background a query, loading the results into a table:

```
>>> from gcloud import bigquery
>>> client = bigquery.Client()
>>> query = """\
SELECT firstname + ' ' + last_name AS full_name,
       FLOOR(DATEDIFF(CURRENT_DATE(), birth_date) / 365) AS age
FROM dataset_name.persons
"""
>>> dataset = client.dataset('dataset_name')
>>> table = dataset.table(name='person_ages')
>>> job = client.run_async_query('fullname-age-query-job', query)
>>> job.destination_table = table
>>> job.write_disposition= 'truncate'
>>> job.name
'fullname-age-query-job'
>>> job.job_type
'query'
>>> job.created
```

```
None
>>> job.state
None
```

Note:

- `gcloud.bigquery` generates a UUID for each job.
 - The `created` and `state` fields are not set until the job is submitted to the BigQuery back-end.
-

Then, begin executing the job on the server:

```
>>> job.submit() # API call
>>> job.created
datetime.datetime(2015, 7, 23, 9, 30, 20, 268260, tzinfo=<UTC>)
>>> job.state
'running'
```

Poll until the job is complete:

```
>>> import time
>>> retry_count = 100
>>> while retry_count > 0 and job.state == 'running':
...     retry_count -= 1
...     time.sleep(10)
...     job.reload() # API call
>>> job.state
'done'
>>> job.ended
datetime.datetime(2015, 7, 23, 9, 30, 21, 334792, tzinfo=<UTC>)
```

24.5.3 Inserting data (synchronous)

Load data synchronously from a local CSV file into a new table:

```
>>> import csv
>>> from gcloud import bigquery
>>> from gcloud.bigquery import SchemaField
>>> client = bigquery.Client()
>>> table = dataset.table(name='person_ages')
>>> table.schema = [
...     SchemaField(name='full_name', type='string', mode='required'),
...     SchemaField(name='age', type='int', mode='required')]
>>> with open('/path/to/person_ages.csv', 'rb') as file_obj:
...     reader = csv.reader(file_obj)
...     rows = list(reader)
>>> table.insert_data(rows) # API request
```

24.5.4 Inserting data (asynchronous)

Start a job loading data asynchronously from a set of CSV files, located on Google Cloud Storage, appending rows into an existing table. First, create the job locally:

```

>>> from gcloud import bigquery
>>> from gcloud.bigquery import SchemaField
>>> client = bigquery.Client()
>>> table = dataset.table(name='person_ages')
>>> table.schema = [
...     SchemaField(name='full_name', type='string', mode='required'),
...     SchemaField(name='age', type='int', mode='required')]
>>> job = client.load_table_from_storage(
...     'load-from-storage-job', table, 'gs://bucket-name/object-prefix*')
>>> job.source_format = 'CSV'
>>> job.skip_leading_rows = 1 # count of skipped header rows
>>> job.write_disposition = 'truncate'
>>> job.name
'load-from-storage-job'
>>> job.job_type
'load'
>>> job.created
None
>>> job.state
None

```

Note:

- `gcloud.bigquery` generates a UUID for each job.
- The `created` and `state` fields are not set until the job is submitted to the BigQuery back-end.

Then, begin executing the job on the server:

```

>>> job.begin() # API call
>>> job.created
datetime.datetime(2015, 7, 23, 9, 30, 20, 268260, tzinfo=<UTC>)
>>> job.state
'running'

```

Poll until the job is complete:

```

>>> import time
>>> retry_count = 100
>>> while retry_count > 0 and job.state == 'running':
...     retry_count -= 1
...     time.sleep(10)
...     job.reload() # API call
>>> job.state
'done'
>>> job.ended
datetime.datetime(2015, 7, 23, 9, 30, 21, 334792, tzinfo=<UTC>)

```

24.5.5 Exporting data (async)

Start a job exporting a table's data asynchronously to a set of CSV files, located on Google Cloud Storage. First, create the job locally:

```

>>> from gcloud import bigquery
>>> client = bigquery.Client()
>>> table = dataset.table(name='person_ages')

```

```
>>> job = client.extract_table_to_storage(
...     'extract-person-ages-job', table,
...     'gs://bucket-name/export-prefix*.csv')
... job.destination_format = 'CSV'
... job.print_header = True
... job.write_disposition = 'truncate'
>>> job.name
'extract-person-ages-job'
>>> job.job_type
'extract'
>>> job.created
None
>>> job.state
None
```

Note:

- `gcloud.bigquery` generates a UUID for each job.
 - The `created` and `state` fields are not set until the job is submitted to the BigQuery back-end.
-

Then, begin executing the job on the server:

```
>>> job.begin() # API call
>>> job.created
datetime.datetime(2015, 7, 23, 9, 30, 20, 268260, tzinfo=<UTC>)
>>> job.state
'running'
```

Poll until the job is complete:

```
>>> import time
>>> retry_count = 100
>>> while retry_count > 0 and job.state == 'running':
...     retry_count -= 1
...     time.sleep(10)
...     job.reload() # API call
>>> job.state
'done'
>>> job.ended
datetime.datetime(2015, 7, 23, 9, 30, 21, 334792, tzinfo=<UTC>)
```

24.5.6 Copy tables (async)

First, create the job locally:

```
>>> from gcloud import bigquery
>>> client = bigquery.Client()
>>> source_table = dataset.table(name='person_ages')
>>> destination_table = dataset.table(name='person_ages_copy')
>>> job = client.copy_table(
...     'copy-table-job', destination_table, source_table)
>>> job.name
'copy-table-job'
>>> job.job_type
'copy'
```

```
>>> job.created
None
>>> job.state
None
```

Note:

- `gcloud.bigquery` generates a UUID for each job.
 - The `created` and `state` fields are not set until the job is submitted to the BigQuery back-end.
-

Then, begin executing the job on the server:

```
>>> job.begin() # API call
>>> job.created
datetime.datetime(2015, 7, 23, 9, 30, 20, 268260, tzinfo=<UTC>)
>>> job.state
'running'
```

Poll until the job is complete:

```
>>> import time
>>> retry_count = 100
>>> while retry_count > 0 and job.state == 'running':
...     retry_count -= 1
...     time.sleep(10)
...     job.reload() # API call
>>> job.state
'done'
>>> job.ended
datetime.datetime(2015, 7, 23, 9, 30, 21, 334792, tzinfo=<UTC>)
```

BigQuery Client

Client for interacting with the Google BigQuery API.

class `gcloud.bigquery.client.Client` (*project=None, credentials=None, http=None*)
 Bases: `gcloud.client.JSONClient`

Client to bundle configuration needed for API requests.

Parameters

- **project** (*string*) – the project which the client acts on behalf of. Will be passed when creating a dataset/job. If not passed, falls back to the default inferred from the environment.
- **credentials** (`oauth2client.client.OAuth2Credentials` or `NoneType`) – The OAuth2 Credentials to use for the connection owned by this client. If not passed (and if no `http` object is passed), falls back to the default inferred from the environment.
- **http** (`httplib2.Http` or class that defines `request()`) – An optional HTTP object to make requests. If not passed, an `http` object is created that is bound to the `credentials` for the current object.

copy_table (*job_name, destination, *sources*)

Construct a job for copying one or more tables into another table.

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs#configuration.copy>

Parameters

- **job_name** (*string*) – Name of the job.
- **destination** (`gcloud.bigquery.table.Table`) – Table into which data is to be copied.
- **sources** (sequence of `gcloud.bigquery.table.Table`) – tables to be copied.

Return type `gcloud.bigquery.job.CopyJob`

Returns a new `CopyJob` instance

dataset (*dataset_name*)

Construct a dataset bound to this client.

Parameters **dataset_name** (*string*) – Name of the dataset.

Return type `gcloud.bigquery.dataset.Dataset`

Returns a new `Dataset` instance

extract_table_to_storage (*job_name*, *source*, **destination_uris*)

Construct a job for extracting a table into Cloud Storage files.

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs#configuration.extract>

Parameters

- **job_name** (*string*) – Name of the job.
- **source** (*gcloud.bigquery.table.Table*) – table to be extracted.
- **destination_uris** (*sequence of string*) – URIs of Cloud-Storage file(s) into which table data is to be extracted; in format `gs://<bucket_name>/<object_name_or_glob>`.

Return type *gcloud.bigquery.job.ExtractTableToStorageJob*

Returns a new *ExtractTableToStorageJob* instance

job_from_resource (*resource*)

Detect correct job type from resource and instantiate.

Parameters **resource** (*dict*) – one job resource from API response

Rtype; One of *gcloud.bigquery.job.LoadTableFromStorageJob*,
gcloud.bigquery.job.CopyJob, *gcloud.bigquery.job.ExtractTableToStorageJob*,
gcloud.bigquery.job.QueryJob, *gcloud.bigquery.job.RunSyncQueryJob*

Returns the job instance, constructed via the resource

list_datasets (*include_all=False*, *max_results=None*, *page_token=None*)

List datasets for the project associated with this client.

See: <https://cloud.google.com/bigquery/docs/reference/v2/datasets/list>

Parameters

- **include_all** (*boolean*) – True if results include hidden datasets.
- **max_results** (*int*) – maximum number of datasets to return, If not passed, defaults to a value set by the API.
- **page_token** (*string*) – opaque marker for the next “page” of datasets. If not passed, the API will return the first page of datasets.

Return type tuple, (list, str)

Returns list of *gcloud.bigquery.dataset.Dataset*, plus a “next page token” string: if the token is not None, indicates that more datasets can be retrieved with another call (pass that value as *page_token*).

list_jobs (*max_results=None*, *page_token=None*, *all_users=None*, *state_filter=None*)

List jobs for the project associated with this client.

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs/list>

Parameters

- **max_results** (*int*) – maximum number of jobs to return, If not passed, defaults to a value set by the API.
- **page_token** (*string*) – opaque marker for the next “page” of jobs. If not passed, the API will return the first page of jobs.
- **all_users** (*boolean*) – if true, include jobs owned by all users in the project.

- **state_filter** (*string*) – if passed, include only jobs matching the given state. One of
 - "done"
 - "pending"
 - "running"

Return type tuple, (list, str)

Returns list of job instances, plus a “next page token” string: if the token is not None, indicates that more jobs can be retrieved with another call, passing that value as `page_token`).

load_table_from_storage (*job_name*, *destination*, **source_uris*)

Construct a job for loading data into a table from CloudStorage.

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs#configuration.load>

Parameters

- **job_name** (*string*) – Name of the job.
- **destination** (*gcloud.bigquery.table.Table*) – Table into which data is to be loaded.
- **source_uris** (*sequence of string*) – URIs of data files to be loaded; in format `gs://<bucket_name>/<object_name_or_glob>`.

Return type *gcloud.bigquery.job.LoadTableFromStorageJob*

Returns a new *LoadTableFromStorageJob* instance

run_async_query (*job_name*, *query*)

Construct a job for running a SQL query asynchronously.

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs#configuration.query>

Parameters

- **job_name** (*string*) – Name of the job.
- **query** (*string*) – SQL query to be executed

Return type *gcloud.bigquery.job.QueryJob*

Returns a new *QueryJob* instance

run_sync_query (*query*)

Run a SQL query synchronously.

Parameters **query** (*string*) – SQL query to be executed

Return type *gcloud.bigquery.query.QueryResults*

Returns a new *QueryResults* instance

25.1 Connection

Create / interact with gcloud bigquery connections.

class `gcloud.bigquery.connection.Connection` (*credentials=None*, *http=None*)

Bases: *gcloud.connection.JSONConnection*

A connection to Google Cloud BigQuery via the JSON REST API.

API_BASE_URL = 'https://www.googleapis.com'

The base of the API call URL.

API_URL_TEMPLATE = '{api_base_url}/bigquery/{api_version}{path}'

A template for the URL of a particular API call.

API_VERSION = 'v2'

The version of the API, used in building the API call's URL.

SCOPE = ('https://www.googleapis.com/auth/bigquery', 'https://www.googleapis.com/auth/cloud-platform')

The scopes required for authenticating as a Cloud BigQuery consumer.

Datasets

Define API Datasets.

class `gcloud.bigquery.dataset.AccessGrant` (*role, entity_type, entity_id*)
 Bases: `object`

Represent grant of an access role to an entity.

Every entry in the access list will have exactly one of `userByEmail`, `groupByEmail`, `domain`, `specialGroup` or `view` set. And if anything but `view` is set, it'll also have a `role` specified. `role` is omitted for a view, since views are always read-only.

See <https://cloud.google.com/bigquery/docs/reference/v2/datasets>.

Parameters

- **role** (*string*) – Role granted to the entity. One of
 - 'OWNER'
 - 'WRITER'
 - 'READER'
 May also be `None` if the `entity_type` is `view`.
- **entity_type** (*string*) – Type of entity being granted the role. One of `ENTITY_TYPES`.
- **entity_id** (*string*) – ID of entity being granted the role.

Raises `ValueError` if the `entity_type` is not among `ENTITY_TYPES`, or if a `view` has `role` set or a non `view` **does not** have a `role` set.

ENTITY_TYPES = `frozenset(['specialGroup', 'groupByEmail', 'userByEmail', 'domain', 'view'])`
 Allowed entity types.

class `gcloud.bigquery.dataset.Dataset` (*name, client, access_grants=()*)
 Bases: `object`

Datasets are containers for tables.

See: <https://cloud.google.com/bigquery/docs/reference/v2/datasets>

Parameters

- **name** (*string*) – the name of the dataset
- **client** (*gcloud.bigquery.client.Client*) – A client which holds credentials and project configuration for the dataset (which requires a project).

- **access_grants** (list of *AccessGrant*) – roles granted to entities for this dataset

access_grants

Dataset's access grants.

Return type list of *AccessGrant*

Returns roles granted to entities for this dataset

create (*client=None*)

API call: create the dataset via a PUT request

See: <https://cloud.google.com/bigquery/docs/reference/v2/tables/insert>

Parameters **client** (*gcloud.bigquery.client.Client* or *NoneType*) – the client to use. If not passed, falls back to the `client` stored on the current dataset.

created

Datetime at which the dataset was created.

Return type *datetime.datetime*, or *NoneType*

Returns the creation time (None until set from the server).

dataset_id

ID for the dataset resource.

Return type string, or *NoneType*

Returns the ID (None until set from the server).

default_table_expiration_ms

Default expiration time for tables in the dataset.

Return type integer, or *NoneType*

Returns The time in milliseconds, or None (the default).

delete (*client=None*)

API call: delete the dataset via a DELETE request

See: <https://cloud.google.com/bigquery/docs/reference/v2/tables/delete>

Parameters **client** (*gcloud.bigquery.client.Client* or *NoneType*) – the client to use. If not passed, falls back to the `client` stored on the current dataset.

description

Description of the dataset.

Return type string, or *NoneType*

Returns The description as set by the user, or None (the default).

etag

ETag for the dataset resource.

Return type string, or *NoneType*

Returns the ETag (None until set from the server).

exists (*client=None*)

API call: test for the existence of the dataset via a GET request

See <https://cloud.google.com/bigquery/docs/reference/v2/datasets/get>

Parameters **client** (*gcloud.bigquery.client.Client* or *NoneType*) – the client to use. If not passed, falls back to the `client` stored on the current dataset.

friendly_name

Title of the dataset.

Return type string, or `NoneType`

Returns The name as set by the user, or `None` (the default).

classmethod from_api_repr (*resource*, *client*)

Factory: construct a dataset given its API representation

Parameters

- **resource** (*dict*) – dataset resource representation returned from the API
- **client** (*gcloud.bigquery.client.Client*) – Client which holds credentials and project configuration for the dataset.

Return type *gcloud.bigquery.dataset.Dataset*

Returns Dataset parsed from *resource*.

list_tables (*max_results=None*, *page_token=None*)

List tables for the project associated with this client.

See: <https://cloud.google.com/bigquery/docs/reference/v2/tables/list>

Parameters

- **max_results** (*int*) – maximum number of tables to return, If not passed, defaults to a value set by the API.
- **page_token** (*string*) – opaque marker for the next “page” of datasets. If not passed, the API will return the first page of datasets.

Return type tuple, (list, str)

Returns list of *gcloud.bigquery.table.Table*, plus a “next page token” string: if not `None`, indicates that more tables can be retrieved with another call (pass that value as *page_token*).

location

Location in which the dataset is hosted.

Return type string, or `NoneType`

Returns The location as set by the user, or `None` (the default).

modified

Datetime at which the dataset was last modified.

Return type *datetime.datetime*, or `NoneType`

Returns the modification time (`None` until set from the server).

patch (*client=None*, ***kw*)

API call: update individual dataset properties via a PATCH request

See <https://cloud.google.com/bigquery/docs/reference/v2/datasets/patch>

Parameters

- **client** (*gcloud.bigquery.client.Client* or `NoneType`) – the client to use. If not passed, falls back to the *client* stored on the current dataset.
- **kw** (*dict*) – properties to be patched.

Raises `ValueError` for invalid value types.

path

URL path for the dataset's APIs.

Return type `string`

Returns the path based on project and dataset name.

project

Project bound to the dataset.

Return type `string`

Returns the project (derived from the client).

reload (*client=None*)

API call: refresh dataset properties via a GET request

See <https://cloud.google.com/bigquery/docs/reference/v2/datasets/get>

Parameters **client** (*gcloud.bigquery.client.Client* or `NoneType`) – the client to use. If not passed, falls back to the `client` stored on the current dataset.

self_link

URL for the dataset resource.

Return type `string`, or `NoneType`

Returns the URL (None until set from the server).

table (*name, schema=()*)

Construct a table bound to this dataset.

Parameters

- **name** (*string*) – Name of the table.
- **schema** (list of *gcloud.bigquery.table.SchemaField*) – The table's schema

Return type *gcloud.bigquery.table.Table*

Returns a new `Table` instance

update (*client=None*)

API call: update dataset properties via a PUT request

See <https://cloud.google.com/bigquery/docs/reference/v2/datasets/update>

Parameters **client** (*gcloud.bigquery.client.Client* or `NoneType`) – the client to use. If not passed, falls back to the `client` stored on the current dataset.

Jobs

Define API Jobs.

```
class gcloud.bigquery.job.Compression(name)
    Bases: gcloud.bigquery._helpers._EnumProperty
```

Pseudo-enum for compression properties.

```
ALLOWED = ('GZIP', 'NONE')
```

```
GZIP = 'GZIP'
```

```
NONE = 'NONE'
```

```
class gcloud.bigquery.job.CopyJob(name, destination, sources, client)
    Bases: gcloud.bigquery.job._AsyncJob
```

Asynchronous job: copy data into a table from other tables.

Parameters

- **name** (*string*) – the name of the job
- **destination** (*gcloud.bigquery.table.Table*) – Table into which data is to be loaded.
- **sources** (list of *gcloud.bigquery.table.Table*) – Table into which data is to be loaded.
- **client** (*gcloud.bigquery.client.Client*) – A client which holds credentials and project configuration for the dataset (which requires a project).

create_disposition

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs#configuration.copy.createDisposition>

```
classmethod from_api_repr(resource, client)
```

Factory: construct a job given its API representation

Parameters

- **resource** (*dict*) – dataset job representation returned from the API
- **client** (*gcloud.bigquery.client.Client*) – Client which holds credentials and project configuration for the dataset.

Return type *gcloud.bigquery.job.CopyJob*

Returns Job parsed from *resource*.

write_disposition

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs#configuration.copy.writeDisposition>

class `gcloud.bigquery.job.CreateDisposition` (*name*)

Bases: `gcloud.bigquery._helpers._EnumProperty`

Pseudo-enum for `create_disposition` properties.

ALLOWED = ('CREATE_IF_NEEDED', 'CREATE_NEVER')

CREATE_IF_NEEDED = 'CREATE_IF_NEEDED'

CREATE_NEVER = 'CREATE_NEVER'

class `gcloud.bigquery.job.DestinationFormat` (*name*)

Bases: `gcloud.bigquery._helpers._EnumProperty`

Pseudo-enum for `destination_format` properties.

ALLOWED = ('CSV', 'NEWLINE_DELIMITED_JSON', 'AVRO')

AVRO = 'AVRO'

CSV = 'CSV'

NEWLINE_DELIMITED_JSON = 'NEWLINE_DELIMITED_JSON'

class `gcloud.bigquery.job.Encoding` (*name*)

Bases: `gcloud.bigquery._helpers._EnumProperty`

Pseudo-enum for encoding properties.

ALLOWED = ('UTF-8', 'ISO-8559-1')

ISO_8559_1 = 'ISO-8559-1'

UTF_8 = 'UTF-8'

class `gcloud.bigquery.job.ExtractTableToStorageJob` (*name*, *source*, *destination_uris*, *client*)

Bases: `gcloud.bigquery.job._AsyncJob`

Asynchronous job: extract data from a table into Cloud Storage.

Parameters

- **name** (*string*) – the name of the job
- **source** (`gcloud.bigquery.table.Table`) – Table into which data is to be loaded.
- **destination_uris** (*list of string*) – URIs describing Cloud Storage blobs into which extracted data will be written, in format `gs://<bucket_name>/<object_name_or_glob>`.
- **client** (`gcloud.bigquery.client.Client`) – A client which holds credentials and project configuration for the dataset (which requires a project).

compression

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs#configuration.extracted.compression>

destination_format

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs#configuration.extracted.destinationFormat>

field_delimiter

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs#configuration.extracted.fieldDelimiter>

classmethod `from_api_repr` (*resource*, *client*)

Factory: construct a job given its API representation

Parameters

- **resource** (*dict*) – dataset job representation returned from the API
- **client** (*gcloud.bigquery.client.Client*) – Client which holds credentials and project configuration for the dataset.

Return type *gcloud.bigquery.job.ExtractTableToStorageJob*

Returns Job parsed from *resource*.

print_header

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs#configuration.extracted.printHeader>

class *gcloud.bigquery.job.LoadTableFromStorageJob* (*name, destination, source_uris, client, schema=()*)

Bases: *gcloud.bigquery.job._AsyncJob*

Asynchronous job for loading data into a table from CloudStorage.

Parameters

- **name** (*string*) – the name of the job
- **destination** (*gcloud.bigquery.table.Table*) – Table into which data is to be loaded.
- **source_uris** (*sequence of string*) – URIs of one or more data files to be loaded, in format *gs://<bucket_name>/<object_name_or_glob>*.
- **client** (*gcloud.bigquery.client.Client*) – A client which holds credentials and project configuration for the dataset (which requires a project).
- **schema** (list of *gcloud.bigquery.table.SchemaField*) – The job's schema

allow_jagged_rows

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs#configuration.load.allowJaggedRows>

allow_quoted_newlines

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs#configuration.load.allowQuotedNewlines>

create_disposition

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs#configuration.load.createDisposition>

encoding

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs#configuration.load.encoding>

field_delimiter

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs#configuration.load.fieldDelimiter>

classmethod *from_api_repr* (*resource, client*)

Factory: construct a job given its API representation

Parameters

- **resource** (*dict*) – dataset job representation returned from the API
- **client** (*gcloud.bigquery.client.Client*) – Client which holds credentials and project configuration for the dataset.

Return type *gcloud.bigquery.job.LoadTableFromStorageJob*

Returns Job parsed from *resource*.

ignore_unknown_values

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs#configuration.load.ignoreUnknownValues>

input_file_bytes

Count of bytes loaded from source files.

Return type integer, or `NoneType`

Returns the count (None until set from the server).

input_files

Count of source files.

Return type integer, or `NoneType`

Returns the count (None until set from the server).

max_bad_records

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs#configuration.load.maxBadRecords>

output_bytes

Count of bytes saved to destination table.

Return type integer, or `NoneType`

Returns the count (None until set from the server).

output_rows

Count of rows saved to destination table.

Return type integer, or `NoneType`

Returns the count (None until set from the server).

quote_character

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs#configuration.load.quote>

schema

Table's schema.

Return type list of `SchemaField`

Returns fields describing the schema

skip_leading_rows

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs#configuration.load.skipLeadingRows>

source_format

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs#configuration.load.sourceFormat>

write_disposition

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs#configuration.load.writeDisposition>

class `gcloud.bigquery.job.QueryJob` (*name, query, client*)

Bases: `gcloud.bigquery.job._AsyncJob`

Asynchronous job: query tables.

Parameters

- **name** (*string*) – the name of the job
- **query** (*string*) – SQL query string
- **client** (`gcloud.bigquery.client.Client`) – A client which holds credentials and project configuration for the dataset (which requires a project).

allow_large_results

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs#configuration.query.allowLargeResults>

create_disposition

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs#configuration.query.createDisposition>

default_dataset

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs#configuration.query.defaultDataset>

destination

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs#configuration.query.destinationTable>

flatten_results

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs#configuration.query.flattenResults>

classmethod from_api_repr (*resource*, *client*)

Factory: construct a job given its API representation

Parameters

- **resource** (*dict*) – dataset job representation returned from the API
- **client** (*gcloud.bigquery.client.Client*) – Client which holds credentials and project configuration for the dataset.

Return type `gcloud.bigquery.job.RunAsyncQueryJob`

Returns Job parsed from `resource`.

priority

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs#configuration.query.priority>

use_query_cache

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs#configuration.query.useQueryCache>

write_disposition

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs#configuration.query.writeDisposition>

class `gcloud.bigquery.job.QueryPriority` (*name*)
 Bases: `gcloud.bigquery._helpers._EnumProperty`

Pseudo-enum for `QueryJob.priority` property.

ALLOWED = ('INTERACTIVE', 'BATCH')

BATCH = 'BATCH'

INTERACTIVE = 'INTERACTIVE'

class `gcloud.bigquery.job.SourceFormat` (*name*)
 Bases: `gcloud.bigquery._helpers._EnumProperty`

Pseudo-enum for `source_format` properties.

ALLOWED = ('CSV', 'DATASTORE_BACKUP', 'NEWLINE_DELIMITED_JSON')

CSV = 'CSV'

DATASTORE_BACKUP = 'DATASTORE_BACKUP'

NEWLINE_DELIMITED_JSON = 'NEWLINE_DELIMITED_JSON'

class `gcloud.bigquery.job.WriteDisposition` (*name*)
 Bases: `gcloud.bigquery._helpers._EnumProperty`

Pseudo-enum for `write_disposition` properties.

ALLOWED = ('WRITE_APPEND', 'WRITE_TRUNCATE', 'WRITE_EMPTY')

WRITE_APPEND = 'WRITE_APPEND'

```
WRITE_EMPTY = 'WRITE_EMPTY'
```

```
WRITE_TRUNCATE = 'WRITE_TRUNCATE'
```

Tables

Define API Datasets.

```
class gcloud.bigquery.table.SchemaField(name, field_type, mode='NULLABLE', description=None, fields=None)
```

Bases: `object`

Describe a single field within a table schema.

Parameters

- **name** (*string*) – the name of the field
- **field_type** (*string*) – the type of the field (one of 'STRING', 'INTEGER', 'FLOAT', 'BOOLEAN', 'TIMESTAMP' or 'RECORD')
- **mode** (*string*) – the type of the field (one of 'NULLABLE', 'REQUIRED', or 'REPEATED')
- **description** (*string*) – optional description for the field
- **fields** (list of *SchemaField*, or None) – subfields (requires `field_type` of 'RECORD').

```
class gcloud.bigquery.table.Table(name, dataset, schema=())
```

Bases: `object`

Tables represent a set of rows whose values correspond to a schema.

See: <https://cloud.google.com/bigquery/docs/reference/v2/tables>

Parameters

- **name** (*string*) – the name of the table
- **dataset** (*gcloud.bigquery.dataset.Dataset*) – The dataset which contains the table.
- **schema** (list of *SchemaField*) – The table's schema

create (*client=None*)

API call: create the dataset via a PUT request

See: <https://cloud.google.com/bigquery/docs/reference/v2/tables/insert>

Parameters **client** (*gcloud.bigquery.client.Client* or `NoneType`) – the client to use. If not passed, falls back to the `client` stored on the current dataset.

created

Datetime at which the table was created.

Return type `datetime.datetime`, or `NoneType`

Returns the creation time (None until set from the server).

dataset_name

Name of dataset containing the table.

Return type `string`

Returns the ID (derived from the dataset).

delete (*client=None*)

API call: delete the table via a DELETE request

See: <https://cloud.google.com/bigquery/docs/reference/v2/tables/delete>

Parameters **client** (*gcloud.bigquery.client.Client* or `NoneType`) – the client to use. If not passed, falls back to the `client` stored on the current dataset.

description

Description of the table.

Return type `string`, or `NoneType`

Returns The description as set by the user, or None (the default).

etag

ETag for the table resource.

Return type `string`, or `NoneType`

Returns the ETag (None until set from the server).

exists (*client=None*)

API call: test for the existence of the table via a GET request

See <https://cloud.google.com/bigquery/docs/reference/v2/tables/get>

Parameters **client** (*gcloud.bigquery.client.Client* or `NoneType`) – the client to use. If not passed, falls back to the `client` stored on the current dataset.

expires

Datetime at which the table will be removed.

Return type `datetime.datetime`, or `NoneType`

Returns the expiration time, or None

fetch_data (*max_results=None, page_token=None, client=None*)

API call: fetch the table data via a GET request

See: <https://cloud.google.com/bigquery/docs/reference/v2/tabledata/list>

Note: This method assumes that its instance's `schema` attribute is up-to-date with the schema as defined on the back-end: if the two schemas are not identical, the values returned may be incomplete. To ensure that the local copy of the schema is up-to-date, call the table's `reload` method.

Parameters

- **max_results** (integer or `NoneType`) – maximum number of rows to return.
- **page_token** (string or `NoneType`) – token representing a cursor into the table's rows.

- **client** (*gcloud.bigquery.client.Client* or `NoneType`) – the client to use. If not passed, falls back to the `client` stored on the current dataset.

Return type tuple

Returns (`row_data`, `total_rows`, `page_token`), where `row_data` is a list of tuples, one per result row, containing only the values; `total_rows` is a count of the total number of rows in the table; and `page_token` is an opaque string which can be used to fetch the next batch of rows (`None` if no further batches can be fetched).

friendly_name

Title of the table.

Return type string, or `NoneType`

Returns The name as set by the user, or `None` (the default).

classmethod from_api_repr (*resource*, *dataset*)

Factory: construct a table given its API representation

Parameters

- **resource** (*dict*) – table resource representation returned from the API
- **dataset** (*gcloud.bigquery.dataset.Dataset*) – The dataset containing the table.

Return type *gcloud.bigquery.table.Table*

Returns Table parsed from `resource`.

insert_data (*rows*, *row_ids=None*, *skip_invalid_rows=None*, *ignore_unknown_values=None*, *client=None*)

API call: insert table data via a POST request

See: <https://cloud.google.com/bigquery/docs/reference/v2/tabledata/insertAll>

Parameters

- **rows** (*list of tuples*) – Row data to be inserted. Each tuple should contain data for each schema field on the current table and in the same order as the schema fields.
- **row_ids** (*list of string*) – Unique ids, one per row being inserted. If not passed, no de-duplication occurs.
- **skip_invalid_rows** (boolean or `NoneType`) – skip rows w/ invalid data?
- **ignore_unknown_values** (boolean or `NoneType`) – ignore columns beyond schema?
- **client** (*gcloud.bigquery.client.Client* or `NoneType`) – the client to use. If not passed, falls back to the `client` stored on the current dataset.

Return type list of mappings

Returns One mapping per row with insert errors: the “index” key identifies the row, and the “errors” key contains a list of the mappings describing one or more problems with the row.

location

Location in which the table is hosted.

Return type string, or `NoneType`

Returns The location as set by the user, or `None` (the default).

modified

Datetime at which the table was last modified.

Return type `datetime.datetime`, or `NoneType`

Returns the modification time (None until set from the server).

num_bytes

The size of the table in bytes.

Return type `integer`, or `NoneType`

Returns the byte count (None until set from the server).

num_rows

The number of rows in the table.

Return type `integer`, or `NoneType`

Returns the row count (None until set from the server).

patch (*client=None, friendly_name=<object object>, description=<object object>, location=<object object>, expires=<object object>, view_query=<object object>, schema=<object object>*)
API call: update individual table properties via a PATCH request

See <https://cloud.google.com/bigquery/docs/reference/v2/tables/patch>

Parameters

- **client** (*gcloud.bigquery.client.Client* or `NoneType`) – the client to use. If not passed, falls back to the `client` stored on the current dataset.
- **friendly_name** (`string` or `NoneType`) – point in time at which the table expires.
- **description** (`string` or `NoneType`) – point in time at which the table expires.
- **location** (`string` or `NoneType`) – point in time at which the table expires.
- **expires** (`datetime.datetime` or `NoneType`) – point in time at which the table expires.
- **view_query** (*string*) – SQL query defining the table as a view
- **schema** (list of *SchemaField*) – fields describing the schema

Raises `ValueError` for invalid value types.

path

URL path for the table's APIs.

Return type `string`

Returns the path based on project and dataset name.

project

Project bound to the table.

Return type `string`

Returns the project (derived from the dataset).

reload (*client=None*)

API call: refresh table properties via a GET request

See <https://cloud.google.com/bigquery/docs/reference/v2/tables/get>

Parameters **client** (*gcloud.bigquery.client.Client* or `NoneType`) – the client to use. If not passed, falls back to the `client` stored on the current dataset.

schema

Table's schema.

Return type list of *SchemaField*

Returns fields describing the schema

self_link

URL for the table resource.

Return type string, or *NoneType*

Returns the URL (None until set from the server).

table_id

ID for the table resource.

Return type string, or *NoneType*

Returns the ID (None until set from the server).

table_type

The type of the table.

Possible values are "TABLE" or "VIEW".

Return type string, or *NoneType*

Returns the URL (None until set from the server).

update (*client=None*)

API call: update table properties via a PUT request

See <https://cloud.google.com/bigquery/docs/reference/v2/tables/update>

Parameters **client** (*gcloud.bigquery.client.Client* or *NoneType*) – the client to use. If not passed, falls back to the `client` stored on the current dataset.

upload_from_file (*file_obj*, *source_format*, *rewind=False*, *size=None*, *num_retries=6*, *allow_jagged_rows=None*, *allow_quoted_newlines=None*, *create_disposition=None*, *encoding=None*, *field_delimiter=None*, *ignore_unknown_values=None*, *max_bad_records=None*, *quote_character=None*, *skip_leading_rows=None*, *write_disposition=None*, *client=None*)

Upload the contents of this table from a file-like object.

The content type of the upload will either be - The value passed in to the function (if any) - `text/csv`.

Parameters

- **file_obj** (*file*) – A file handle open for reading.
- **source_format** (*string*) – one of 'CSV' or 'NEWLINE_DELIMITED_JSON'. job configuration option; see `gcloud.bigquery.job.LoadJob()`
- **rewind** (*boolean*) – If True, seek to the beginning of the file handle before writing the file to Cloud Storage.
- **size** (*int*) – The number of bytes to read from the file handle. If not provided, we'll try to guess the size using `os.fstat()`. (If the file handle is not from the filesystem this won't be possible.)
- **num_retries** (*integer*) – Number of upload retries. Defaults to 6.
- **allow_jagged_rows** (*boolean*) – job configuration option; see `gcloud.bigquery.job.LoadJob()`

- **allow_quoted_newlines** (*boolean*) – job configuration option; see `gcloud.bigquery.job.LoadJob()`
- **create_disposition** (*string*) – job configuration option; see `gcloud.bigquery.job.LoadJob()`
- **encoding** (*string*) – job configuration option; see `gcloud.bigquery.job.LoadJob()`
- **field_delimiter** (*string*) – job configuration option; see `gcloud.bigquery.job.LoadJob()`
- **ignore_unknown_values** (*boolean*) – job configuration option; see `gcloud.bigquery.job.LoadJob()`
- **max_bad_records** (*integer*) – job configuration option; see `gcloud.bigquery.job.LoadJob()`
- **quote_character** (*string*) – job configuration option; see `gcloud.bigquery.job.LoadJob()`
- **skip_leading_rows** (*integer*) – job configuration option; see `gcloud.bigquery.job.LoadJob()`
- **write_disposition** (*string*) – job configuration option; see `gcloud.bigquery.job.LoadJob()`
- **client** (`gcloud.storage.client.Client` or `NoneType`) – Optional. The client to use. If not passed, falls back to the `client` stored on the current dataset.

Return type `gcloud.bigquery.jobs.LoadTableFromStorageJob`

Returns the job instance used to load the data (e.g., for querying status)

Raises `ValueError` if size is not passed in and can not be determined

view_query

SQL query defining the table as a view.

Return type `string`, or `NoneType`

Returns The query as set by the user, or `None` (the default).

Query

Define API Queries.

class `gcloud.bigquery.query.QueryResults` (*query*, *client*)

Bases: `object`

Synchronous job: query tables.

Parameters

- **query** (*string*) – SQL query string
- **client** (`gcloud.bigquery.client.Client`) – A client which holds credentials and project configuration for the dataset (which requires a project).

cache_hit

Query results served from cache.

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs/query#cacheHit>

Return type `boolean` or `NoneType`

Returns True if the query results were served from cache (None until set by the server).

complete

Server completed query.

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs/query#jobComplete>

Return type `boolean` or `NoneType`

Returns True if the query completed on the server (None until set by the server).

default_dataset

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs/query#defaultDataset>

errors

Errors generated by the query.

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs/query#errors>

Return type list of mapping, or `NoneType`

Returns Mappings describing errors generated on the server (None until set by the server).

fetch_data (*max_results=None*, *page_token=None*, *start_index=None*, *timeout_ms=None*, *client=None*)

API call: fetch a page of query result data via a GET request

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs/getQueryResults>

Parameters

- **max_results** (integer or `NoneType`) – maximum number of rows to return.
- **page_token** (string or `NoneType`) – token representing a cursor into the table's rows.
- **start_index** (integer or `NoneType`) – zero-based index of starting row
- **timeout_ms** (integer or `NoneType`) – timeout, in milliseconds, to wait for query to complete
- **client** (`gcloud.bigquery.client.Client` or `NoneType`) – the client to use. If not passed, falls back to the `client` stored on the current dataset.

Return type tuple

Returns (`row_data`, `total_rows`, `page_token`), where `row_data` is a list of tuples, one per result row, containing only the values; `total_rows` is a count of the total number of rows in the table; and `page_token` is an opaque string which can be used to fetch the next batch of rows (`None` if no further batches can be fetched).

Raises `ValueError` if the query has not yet been executed.

job

Job instance used to run the query.

Return type `gcloud.bigquery.job.QueryJob`, or `NoneType`

Returns Job instance used to run the query (`None` until `jobReference` property is set by the server).

max_results

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs/query#maxResults>

name

Job name, generated by the back-end.

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs/query#jobReference>

Return type list of mapping, or `NoneType`

Returns Mappings describing errors generated on the server (`None` until set by the server).

page_token

Token for fetching next batch of results.

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs/query#pageToken>

Return type string, or `NoneType`

Returns Token generated on the server (`None` until set by the server).

preserve_nulls

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs/query#preserveNulls>

project

Project bound to the job.

Return type string

Returns the project (derived from the client).

rows

Query results.

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs/query#rows>

Return type list of tuples of row values, or `NoneType`

Returns fields describing the schema (None until set by the server).

run (*client=None*)

API call: run the query via a POST request

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs/query>

Parameters **client** (*gcloud.bigquery.client.Client* or `NoneType`) – the client to use. If not passed, falls back to the `client` stored on the current dataset.

schema

Schema for query results.

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs/query#schema>

Return type list of `SchemaField`, or `NoneType`

Returns fields describing the schema (None until set by the server).

timeout_ms

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs/query#timeoutMs>

total_bytes_processed

Total number of bytes processed by the query

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs/query#totalBytesProcessed>

Return type integer, or `NoneType`

Returns Count generated on the server (None until set by the server).

total_rows

Total number of rows returned by the query

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs/query#totalRows>

Return type integer, or `NoneType`

Returns Count generated on the server (None until set by the server).

use_query_cache

See: <https://cloud.google.com/bigquery/docs/reference/v2/jobs/query#useQueryCache>

Resource Manager Overview

The Cloud Resource Manager API provides methods that you can use to programmatically manage your projects in the Google Cloud Platform. With this API, you can do the following:

- Get a list of all projects associated with an account
- Create new projects
- Update existing projects
- Delete projects
- Undelete, or recover, projects that you don't want to delete

Note: Don't forget to look at the *Authentication* section below. It's slightly different from the rest of this library.

Here's a quick example of the full life-cycle:

```
>>> from gcloud import resource_manager
>>> client = resource_manager.Client()

>>> # List all projects you have access to
>>> for project in client.list_projects():
...     print(project)

>>> # Create a new project
>>> new_project = client.new_project('your-project-id-here',
...                                 name='My new project')
>>> new_project.create()

>>> # Update an existing project
>>> project = client.fetch_project('my-existing-project')
>>> print(project)
<Project: Existing Project (my-existing-project)>
>>> project.name = 'Modified name'
>>> project.update()
>>> print(project)
<Project: Modified name (my-existing-project)>

>>> # Delete a project
>>> project = client.new_project('my-existing-project')
>>> project.delete()

>>> # Undelete a project
```

```
>>> project = client.new_project('my-existing-project')
>>> project.undelete()
```

30.1 Authentication

Unlike the other APIs, the Resource Manager API is focused on managing your various projects inside Google Cloud Platform. What this means (currently, as of August 2015) is that you can't use a Service Account to work with some parts of this API (for example, creating projects).

The reason is actually pretty simple: if your API call is trying to do something like create a project, what project's Service Account can you use? Currently none.

This means that for this API you should always use the credentials provided by the [Google Cloud SDK](#), which you can get by running `gcloud auth login`.

Once you run that command, `gcloud-python` will automatically pick up the credentials, and you can use the “automatic discovery” feature of the library.

Start by authenticating:

```
$ gcloud auth login
```

And then simply create a client:

```
>>> from gcloud import resource_manager
>>> client = resource_manager.Client()
```

Client

A Client for interacting with the Resource Manager API.

class `gcloud.resource_manager.client.Client` (*credentials=None, http=None*)
 Bases: `gcloud.client.Client`

Client to bundle configuration needed for API requests.

See <https://cloud.google.com/resource-manager/reference/rest/> for more information on this API.

Automatically get credentials:

```
>>> from gcloud import resource_manager
>>> client = resource_manager.Client()
```

Parameters

- **credentials** (`oauth2client.client.OAuth2Credentials` or `NoneType`) – The OAuth2 Credentials to use for the connection owned by this client. If not passed (and if no `http` object is passed), falls back to the default inferred from the environment.
- **http** (`httplib2.Http` or class that defines `request()`) – An optional HTTP object to make requests. If not passed, an `http` object is created that is bound to the `credentials` for the current object.

fetch_project (*project_id*)

Fetch an existing project and its relevant metadata by ID.

Note: If the project does not exist, this will raise a *NotFound* error.

Parameters `project_id` (*str*) – The ID for this project.

Return type *Project*

Returns A *Project* with metadata fetched from the API.

list_projects (*filter_params=None, page_size=None*)

List the projects visible to this client.

Example:

```
>>> from gcloud import resource_manager
>>> client = resource_manager.Client()
>>> for project in client.list_projects():
...     print project.project_id
```

List all projects with label 'environment' set to 'prod' (filtering by labels):

```
>>> from gcloud import resource_manager
>>> client = resource_manager.Client()
>>> env_filter = {'labels.environment': 'prod'}
>>> for project in client.list_projects(env_filter):
...     print project.project_id
```

See: <https://cloud.google.com/resource-manager/reference/rest/v1beta1/projects/list>

Complete filtering example:

```
>>> project_filter = { # Return projects with...
...     'name': 'My Project', # name set to 'My Project'.
...     'id': 'my-project-id', # id set to 'my-project-id'.
...     'labels.stage': 'prod', # the label 'stage' set to 'prod'
...     'labels.color': '*' # a label 'color' set to anything.
... }
>>> client.list_projects(project_filter)
```

Parameters

- **filter_params** (*dict*) – (Optional) A dictionary of filter options where each key is a property to filter on, and each value is the (case-insensitive) value to check (or the glob * to check for existence of the property). See the example above for more details.
- **page_size** (*int*) – (Optional) Maximum number of projects to return in a single page. If not passed, defaults to a value set by the API.

Return type `_ProjectIterator`

Returns A project iterator. The iterator will make multiple API requests if you continue iterating and there are more pages of results. Each item returned will be a `Project`.

new_project (*project_id*, *name=None*, *labels=None*)

Creates a `Project` bound to the current client.

Use `Project.reload()` to retrieve project metadata after creating a `Project` instance.

Parameters

- **project_id** (*str*) – The ID for this project.
- **name** (*string*) – The display name of the project.
- **labels** (*dict*) – A list of labels associated with the project.

Return type `Project`

Returns A new instance of a `Project` **without** any metadata loaded.

31.1 Connection

Create / interact with `gcloud.resource_manager` connections.

class `gcloud.resource_manager.connection.Connection` (*credentials=None, http=None*)
Bases: `gcloud.connection.JSONConnection`

A connection to Google Cloud Resource Manager via the JSON REST API.

Parameters

- **credentials** (`oauth2client.client.OAuth2Credentials`) – (Optional) The OAuth2 Credentials to use for this connection.
- **http** (`httplib2.Http` or class that defines `request()`) – (Optional) HTTP object to make requests.

API_BASE_URL = `'https://cloudresourcemanager.googleapis.com'`

The base of the API call URL.

API_URL_TEMPLATE = `'{api_base_url}/{api_version}{path}'`

A template for the URL of a particular API call.

API_VERSION = `'v1beta1'`

The version of the API, used in building the API call's URL.

SCOPE = `('https://www.googleapis.com/auth/cloud-platform',)`

The scopes required for authenticating as a Resource Manager consumer.

Projects

Utility for managing projects via the Cloud Resource Manager API.

```
class gcloud.resource_manager.project.Project (project_id, client, name=None, labels=None)
```

Bases: `object`

Projects are containers for your work on Google Cloud Platform.

Note: A `Project` can also be created via `Client.new_project()`

To manage labels on a `Project`:

```
>>> from gcloud import resource_manager
>>> client = resource_manager.Client()
>>> project = client.new_project('purple-spaceship-123')
>>> project.labels = {'color': 'purple'}
>>> project.labels['environment'] = 'production'
>>> project.update()
```

See: <https://cloud.google.com/resource-manager/reference/rest/v1beta1/projects>

Parameters

- **project_id** (*string*) – The globally unique ID of the project.
- **client** (*gcloud.resource_manager.client.Client*) – The Client used with this project.
- **name** (*string*) – The display name of the project.
- **labels** (*dict*) – A list of labels associated with the project.

create (*client=None*)

API call: create the project via a POST request.

See <https://cloud.google.com/resource-manager/reference/rest/v1beta1/projects/create>

Parameters **client** (*gcloud.resource_manager.client.Client* or `NoneType`)
– the client to use. If not passed, falls back to the client stored on the current project.

delete (*client=None, reload_data=False*)

API call: delete the project via a DELETE request.

See: <https://cloud.google.com/resource-manager/reference/rest/v1beta1/projects/delete>

This actually changes the status (`lifecycleState`) from `ACTIVE` to `DELETE_REQUESTED`. Later (it's not specified when), the project will move into the `DELETE_IN_PROGRESS` state, which means the deleting has actually begun.

Parameters

- **client** (*gcloud.resource_manager.client.Client* or `NoneType`) – the client to use. If not passed, falls back to the client stored on the current project.
- **reload_data** (*bool*) – Whether to reload the project with the latest state. If you want to get the updated status, you'll want this set to `True` as the `DELETE` method doesn't send back the updated project. Default: `False`.

exists (*client=None*)

API call: test the existence of a project via a `GET` request.

See <https://cloud.google.com/resource-manager/reference/rest/v1beta1/projects/get>

Parameters **client** (*gcloud.resource_manager.client.Client* or `NoneType`) – the client to use. If not passed, falls back to the client stored on the current project.

classmethod from_api_repr (*resource, client*)

Factory: construct a project given its API representation.

Parameters

- **resource** (*dict*) – project resource representation returned from the API
- **client** (*gcloud.resource_manager.client.Client*) – The Client used with this project.

Return type *gcloud.resource_manager.project.Project*

full_name

Fully-qualified name (ie, `'projects/purple-spaceship-123'`).

path

URL for the project (ie, `'/projects/purple-spaceship-123'`).

reload (*client=None*)

API call: reload the project via a `GET` request.

This method will reload the newest metadata for the project. If you've created a new *Project* instance via *Client.new_project()*, this method will retrieve project metadata.

Warning: This will overwrite any local changes you've made and not saved via *update()*.

See <https://cloud.google.com/resource-manager/reference/rest/v1beta1/projects/get>

Parameters **client** (*gcloud.resource_manager.client.Client* or `NoneType`) – the client to use. If not passed, falls back to the client stored on the current project.

set_properties_from_api_repr (*resource*)

Update specific properties from its API representation.

undelelete (*client=None, reload_data=False*)

API call: undelete the project via a `POST` request.

See <https://cloud.google.com/resource-manager/reference/rest/v1beta1/projects/undelelete>

This actually changes the project status (`lifecycleState`) from `DELETE_REQUESTED` to `ACTIVE`. If the project has already reached a status of `DELETE_IN_PROGRESS`, this request will fail and the project cannot be restored.

Parameters

- **client** (*gcloud.resource_manager.client.Client* or *NoneType*) – the client to use. If not passed, falls back to the client stored on the current project.
- **reload_data** (*bool*) – Whether to reload the project with the latest state. If you want to get the updated status, you'll want this set to *True* as the DELETE method doesn't send back the updated project. Default: *False*.

update (*client=None*)

API call: update the project via a PUT request.

See <https://cloud.google.com/resource-manager/reference/rest/v1beta1/projects/update>

Parameters **client** (*gcloud.resource_manager.client.Client* or *NoneType*)
– the client to use. If not passed, falls back to the client stored on the current project.

Using the API

33.1 Client

Client objects provide a means to configure your DNS applications. Each instance holds both a `project` and an authenticated connection to the DNS service.

For an overview of authentication in `gcloud-python`, see [Authentication](#).

Assuming your environment is set up as described in that document, create an instance of *Client*.

```
>>> from gcloud import dns
>>> client = dns.Client()
```

33.2 Projects

A project is the top-level container in the DNS API: it is tied closely to billing, and can provide default access control across all its datasets. If no `project` is passed to the client container, the library attempts to infer a project using the environment (including explicit environment variables, GAE, or GCE).

To override the project inferred from the environment, pass an explicit `project` to the constructor, or to either of the alternative classmethod factories:

```
>>> from gcloud import dns
>>> client = dns.Client(project='PROJECT_ID')
```

33.3 Project Quotas

Query the quotas for a given project:

```
>>> from gcloud import dns
>>> client = dns.Client(project='PROJECT_ID')
>>> quotas = client.quotas() # API request
>>> for key, value in sorted(quotas.items()):
...     print('%s: %s' % (key, value))
managedZones: 10000
resourceRecordsPerRrset: 100
rrsetsPerManagedZone: 10000
rrsetAdditionsPerChange: 100
```

```
rrsetDeletionsPerChange: 100
totalRrdataSizePerChange: 10000
```

33.3.1 Project ACLs

Each project has an access control list granting reader / writer / owner permission to one or more entities. This list cannot be queried or set via the API: it must be managed using the Google Developer Console.

33.4 Managed Zones

A “managed zone” is the container for DNS records for the same DNS name suffix and has a set of name servers that accept and responds to queries:

```
>>> from gcloud import dns
>>> client = dns.Client(project='PROJECT_ID')
>>> zone = client.zone('acme-co', description='Acme Company zone',
...                   dns_name='example.com')

>>> zone.exists() # API request
False
>>> zone.create() # API request
>>> zone.exists() # API request
True
```

List the zones for a given project:

```
>>> from gcloud import dns
>>> client = dns.Client(project='PROJECT_ID')
>>> zones = client.list_zones() # API request
>>> [zone.name for zone in zones]
['acme-co']
```

33.5 Resource Record Sets

Each managed zone exposes a read-only set of resource records:

```
>>> from gcloud import dns
>>> client = dns.Client(project='PROJECT_ID')
>>> zone = client.zone('acme-co')
>>> records, page_token = zone.list_resource_record_sets() # API request
>>> [(record.name, record.type, record.ttl, record.rdatas) for record in records]
[('example.com.', 'SOA', 21600, ['ns-cloud1.googlecomains.com dns-admin.google.com'] 21600 3600
```

Note: The `page_token` returned from `zone.list_resource_record_sets()` will be an opaque string if there are more resources than can be returned in a single request. To enumerate them all, repeat calling `zone.list_resource_record_sets()`, passing the `page_token`, until the token is `None`. E.g.

```
>>> records, page_token = zone.list_resource_record_sets() # API request
>>> while page_token is not None:
...     next_batch, page_token = zone.list_resource_record_sets(
...         page_token=page_token) # API request
...     records.extend(next_batch)
```

33.6 Change requests

Update the resource record set for a zone by creating a change request bundling additions to or deletions from the set.

```
>>> import time
>>> from gcloud import dns
>>> client = dns.Client(project='PROJECT_ID')
>>> zone = client.zone('acme-co')
>>> record = dns.ResourceRecord(name='www.example.com', type='CNAME')
>>> change = zone.change_request(additions=[record])
>>> change.begin() # API request
>>> while change.status != 'done':
...     print('Waiting for change to complete')
...     time.sleep(60) # or whatever interval is appropriate
...     change.reload() # API request
```

List changes made to the resource record set for a given zone:

```
>>> from gcloud import dns
>>> client = dns.Client(project='PROJECT_ID')
>>> zone = client.zone('acme-co')
>>> changes = []
>>> changes, page_token = zone.list_changes() # API request
```

Note: The `page_token` returned from `zone.list_changes()` will be an opaque string if there are more changes than can be returned in a single request. To enumerate them all, repeat calling `zone.list_changes()`, passing the `page_token`, until the token is `None`. E.g.:

```
>>> changes, page_token = zone.list_changes() # API request
>>> while page_token is not None:
...     next_batch, page_token = zone.list_changes(
...         page_token=page_token) # API request
...     changes.extend(next_batch)
```


DNS Client

Client for interacting with the Google Cloud DNS API.

class `gcloud.dns.client.Client` (*project=None, credentials=None, http=None*)
 Bases: `gcloud.client.JSONClient`

Client to bundle configuration needed for API requests.

Parameters

- **project** (*string*) – the project which the client acts on behalf of. Will be passed when creating a zone. If not passed, falls back to the default inferred from the environment.
- **credentials** (`oauth2client.client.OAuth2Credentials` or `NoneType`) – The OAuth2 Credentials to use for the connection owned by this client. If not passed (and if no `http` object is passed), falls back to the default inferred from the environment.
- **http** (`httplib2.Http` or class that defines `request()`) – An optional HTTP object to make requests. If not passed, an `http` object is created that is bound to the `credentials` for the current object.

list_zones (*max_results=None, page_token=None*)

List zones for the project associated with this client.

See: <https://cloud.google.com/dns/api/v1/managedZones/list>

Parameters

- **max_results** (*int*) – maximum number of zones to return, If not passed, defaults to a value set by the API.
- **page_token** (*string*) – opaque marker for the next “page” of zones. If not passed, the API will return the first page of zones.

Return type tuple, (list, str)

Returns list of `gcloud.dns.zone.ManagedZone`, plus a “next page token” string: if the token is not `None`, indicates that more zones can be retrieved with another call (pass that value as `page_token`).

quotas ()

Return DNS quotas for the project associated with this client.

See: <https://cloud.google.com/dns/api/v1/projects/get>

Return type mapping

Returns keys for the mapping correspond to those of the `quota` sub-mapping of the project resource.

zone (*name*, *dns_name*)

Construct a zone bound to this client.

Parameters

- **name** (*string*) – Name of the zone.
- **dns_name** (*string*) – DNS name of the zone.

Return type *gcloud.dns.zone.ManagedZone*

Returns a new `ManagedZone` instance

34.1 Connection

Create / interact with gcloud dns connections.

class `gcloud.dns.connection.Connection` (*credentials=None*, *http=None*)

Bases: *gcloud.connection.JSONConnection*

A connection to Google Cloud DNS via the JSON REST API.

API_BASE_URL = `'https://www.googleapis.com'`

The base of the API call URL.

API_URL_TEMPLATE = `'{api_base_url}/dns/{api_version}{path}'`

A template for the URL of a particular API call.

API_VERSION = `'v1'`

The version of the API, used in building the API call's URL.

SCOPE = `('https://www.googleapis.com/auth/ndev.cloudDNS.readwrite')`

The scopes required for authenticating as a Cloud DNS consumer.

Managed Zones

Define API ManagedZones.

```
class gcloud.dns.zone.ManagedZone (name, dns_name, client)
    Bases: object
```

ManagedZones are containers for DNS resource records.

See: <https://cloud.google.com/dns/api/v1/managedZones>

Parameters

- **name** (*string*) – the name of the zone
- **dns_name** (*string*) – the DNS name of the zone
- **client** (*gcloud.dns.client.Client*) – A client which holds credentials and project configuration for the zone (which requires a project).

changes ()

Construct a change set bound to this zone.

Return type *gcloud.dns.changes.Changes*

Returns a new Changes instance

create (client=None)

API call: create the zone via a PUT request

See: <https://cloud.google.com/dns/api/v1/managedZones/create>

Parameters **client** (*gcloud.dns.client.Client* or *NoneType*) – the client to use.
If not passed, falls back to the `client` stored on the current zone.

created

Datetime at which the zone was created.

Return type *datetime.datetime*, or *NoneType*

Returns the creation time (None until set from the server).

delete (client=None)

API call: delete the zone via a DELETE request

See: <https://cloud.google.com/dns/api/v1/managedZones/delete>

Parameters **client** (*gcloud.dns.client.Client* or *NoneType*) – the client to use.
If not passed, falls back to the `client` stored on the current zone.

description

Description of the zone.

Return type string, or `NoneType`

Returns The description as set by the user, or `None` (the default).

exists (*client=None*)

API call: test for the existence of the zone via a GET request

See <https://cloud.google.com/dns/api/v1/managedZones/get>

Parameters **client** (*gcloud.dns.client.Client* or `NoneType`) – the client to use. If not passed, falls back to the `client` stored on the current zone.

classmethod from_api_repr (*resource, client*)

Factory: construct a zone given its API representation

Parameters

- **resource** (*dict*) – zone resource representation returned from the API
- **client** (*gcloud.dns.client.Client*) – Client which holds credentials and project configuration for the zone.

Return type *gcloud.dns.zone.ManagedZone*

Returns Zone parsed from `resource`.

list_changes (*max_results=None, page_token=None, client=None*)

List change sets for this zone.

See: <https://cloud.google.com/dns/api/v1/resourceRecordSets/list>

Parameters

- **max_results** (*int*) – maximum number of zones to return, If not passed, defaults to a value set by the API.
- **page_token** (*string*) – opaque marker for the next “page” of zones. If not passed, the API will return the first page of zones.
- **client** (*gcloud.dns.client.Client* or `NoneType`) – the client to use. If not passed, falls back to the `client` stored on the current zone.

Return type tuple, (list, str)

Returns list of *gcloud.dns.resource_record_set.ResourceRecordSet*, plus a “next page token” string: if the token is not `None`, indicates that more zones can be retrieved with another call (pass that value as `page_token`).

list_resource_record_sets (*max_results=None, page_token=None, client=None*)

List resource record sets for this zone.

See: <https://cloud.google.com/dns/api/v1/resourceRecordSets/list>

Parameters

- **max_results** (*int*) – maximum number of zones to return, If not passed, defaults to a value set by the API.
- **page_token** (*string*) – opaque marker for the next “page” of zones. If not passed, the API will return the first page of zones.
- **client** (*gcloud.dns.client.Client* or `NoneType`) – the client to use. If not passed, falls back to the `client` stored on the current zone.

Return type tuple, (list, str)

Returns list of `gcloud.dns.resource_record_set.ResourceRecordSet`, plus a “next page token” string: if the token is not None, indicates that more zones can be retrieved with another call (pass that value as `page_token`).

name_server_set

Named set of DNS name servers that all host the same ManagedZones.

Most users will leave this blank.

See: <https://cloud.google.com/dns/api/v1/managedZones#nameServerSet>

Return type string, or NoneType

Returns The name as set by the user, or None (the default).

name_servers

Datetime at which the zone was created.

Return type list of strings, or NoneType.

Returns the assigned name servers (None until set from the server).

path

URL path for the zone’s APIs.

Return type string

Returns the path based on project and dataste name.

project

Project bound to the zone.

Return type string

Returns the project (derived from the client).

reload (*client=None*)

API call: refresh zone properties via a GET request

See <https://cloud.google.com/dns/api/v1/managedZones/get>

Parameters `client` (`gcloud.dns.client.Client` or NoneType) – the client to use.
If not passed, falls back to the `client` stored on the current zone.

resource_record_set (*name, record_type, ttl, rrdatas*)

Construct a resource record set bound to this zone.

Parameters

- **name** (*string*) – Name of the record set.
- **record_type** (*string*) – RR type
- **ttl** (*integer*) – TTL for the RR, in seconds
- **rrdatas** (*list of string*) – resource data for the RR

Return type `gcloud.dns.resource_record_set.ResourceRecordSet`

Returns a new ResourceRecordSet instance

zone_id

ID for the zone resource.

Return type string, or NoneType

Returns the ID (None until set from the server).

Resource Record Sets

Define API ResourceRecordSets.

class `gcloud.dns.resource_record_set.ResourceRecordSet` (*name, record_type, ttl, rrdatas, zone*)

Bases: `object`

ResourceRecordSets are DNS resource records.

RRS are owned by a `gcloud.dns.zone.ManagedZone` instance.

See: <https://cloud.google.com/dns/api/v1/resourceRecordSets>

Parameters

- **name** (*string*) – the name of the record set
- **record_type** (*string*) – the RR type of the zone
- **ttl** (*integer*) – TTL (in seconds) for caching the record sets
- **rrdatas** (*list of string*) – one or more lines containing the resource data
- **zone** (`gcloud.dns.zone.ManagedZone`) – A zone which holds one or more record sets.

classmethod `from_api_repr` (*resource, zone*)

Factory: construct a record set given its API representation

Parameters

- **resource** (*dict*) – record sets representation returned from the API
- **zone** (`gcloud.dns.zone.ManagedZone`) – A zone which holds one or more record sets.

Return type `gcloud.dns.zone.ResourceRecordSet`

Returns RRS parsed from `resource`.

Change Sets

Define API ResourceRecordSets.

class `gcloud.dns.changes.Changes` (*zone*)

Bases: `object`

Changes are bundled additions / deletions of DNS resource records.

Changes are owned by a `gcloud.dns.zone.ManagedZone` instance.

See: <https://cloud.google.com/dns/api/v1/changes>

Parameters `zone` (`gcloud.dns.zone.ManagedZone`) – A zone which holds one or more record sets.

add_record_set (*record_set*)

Append a record set to the ‘additions’ for the change set.

Parameters `record_set` (`gcloud.dns.resource_record_set.ResourceRecordSet`)
– the record set to append

Raises `ValueError` if `record_set` is not of the required type.

additions

Resource record sets to be added to the zone.

Return type sequence of `gcloud.dns.resource_record_set.ResourceRecordSet`.

Returns record sets appended via `add_record_set()`

create (*client=None*)

API call: create the change set via a POST request

See: <https://cloud.google.com/dns/api/v1/changes/create>

Parameters `client` (`gcloud.dns.client.Client` or `NoneType`) – the client to use.
If not passed, falls back to the `client` stored on the current zone.

delete_record_set (*record_set*)

Append a record set to the ‘deletions’ for the change set.

Parameters `record_set` (`gcloud.dns.resource_record_set.ResourceRecordSet`)
– the record set to append

Raises `ValueError` if `record_set` is not of the required type.

deletions

Resource record sets to be deleted from the zone.

Return type sequence of `gcloud.dns.resource_record_set.ResourceRecordSet`.

Returns record sets appended via `delete_record_set()`

exists (*client=None*)

API call: test for the existence of the change set via a GET request

See <https://cloud.google.com/dns/api/v1/changes/get>

Parameters **client** (*gcloud.dns.client.Client* or `NoneType`) – the client to use.

If not passed, falls back to the `client` stored on the current zone.

classmethod **from_api_repr** (*resource, zone*)

Factory: construct a change set given its API representation

Parameters

- **resource** (*dict*) – change set representation returned from the API
- **zone** (*gcloud.dns.zone.ManagedZone*) – A zone which holds zero or more change sets.

Return type *gcloud.dns.changes.Changes*

Returns RRS parsed from `resource`.

name

Name of the change set.

Return type `string` or `NoneType`

Returns Name, as set by the back-end, or `None`.

path

URL path for change set APIs.

Return type `string`

Returns the path based on project, zone, and change set names.

reload (*client=None*)

API call: refresh zone properties via a GET request

See <https://cloud.google.com/dns/api/v1/changes/get>

Parameters **client** (*gcloud.dns.client.Client* or `NoneType`) – the client to use.

If not passed, falls back to the `client` stored on the current zone.

started

Time when the change set was started.

Return type `datetime.datetime` or `NoneType`

Returns Time, as set by the back-end, or `None`.

status

Status of the change set.

Return type `string` or `NoneType`

Returns Status, as set by the back-end, or `None`.

Using the API

38.1 Overview

Cloud Search allows an application to quickly perform full-text and geospatial searches without having to spin up instances and without the hassle of managing and maintaining a search service.

Cloud Search provides a model for indexing documents containing structured data, with documents and indexes saved to a separate persistent store optimized for search operations.

The API supports full text matching on string fields and allows indexing any number of documents in any number of indexes.

38.1.1 Client

Client objects provide a means to configure your Cloud Search applications. Each instance holds both a *project* and an authenticated connection to the Cloud Search service.

For an overview of authentication in `gcloud-python`, see [Authentication](#).

Assuming your environment is set up as described in that document, create an instance of *Client*.

```
>>> from gcloud import search
>>> client = search.Client()
```

38.2 Indexes

Indexes are searchable collections of documents.

List all indexes in the client's project:

```
>>> indexes = client.list_indexes() # API call
>>> for index in indexes:
...     print(index.name)
...     field_names = ', '.join([field.name for field in index.fields])
...     print('- %s' % field_names)
index-name
- field-1, field-2
another-index-name
- field-3
```

Create a new index:

```
>>> new_index = client.index('new-index-name')
```

Note: Indexes cannot be created, updated, or deleted directly on the server: they are derived from the documents which are created “within” them.

38.3 Documents

Create a document instance, which is not yet added to its index on the server:

```
>>> index = client.index('index-id')
>>> document = index.document('document-1')
>>> document.exists() # API call
False
>>> document.rank
None
```

Add one or more fields to the document:

```
>>> field = document.Field('fieldname')
>>> field.add_value('string')
```

Save the document into the index:

```
>>> document.create() # API call
>>> document.exists() # API call
True
>>> document.rank # set by the server
1443648166
```

List all documents in an index:

```
>>> documents = index.list_documents() # API call
>>> [document.id for document in documents]
['document-1']
```

Delete a document from its index:

```
>>> document = index.document('to-be-deleted')
>>> document.exists() # API call
True
>>> document.delete() # API call
>>> document.exists() # API call
False
```

Note: To update a document in place after manipulating its fields or rank, just recreate it: E.g.:

```
>>> document = index.document('document-id')
>>> document.exists() # API call
True
>>> document.rank = 12345
>>> field = document.field('field-name')
>>> field.add_value('christina aguilera')
>>> document.create() # API call
```

38.4 Fields

Fields belong to documents and are the data that actually gets searched.

Each field can have multiple values, which can be of the following types:

- String (Python2 `unicode`, Python3 `str`)
- Number (Python `int` or `float`)
- Timestamp (Python `datetime.datetime`)
- Geovalue (Python tuple, (`float`, `float`))

String values can be tokenized using one of three different types of tokenization, which can be passed when the value is added:

- **Atom** (`atom`) means “don’t tokenize this string”, treat it as one thing to compare against.
- **Text** (`text`) means “treat this string as normal text” and split words apart to be compared against.
- **HTML** (`html`) means “treat this string as HTML”, understanding the tags, and treating the rest of the content like Text.

```
>>> from gcloud import search
>>> client = search.Client()
>>> index = client.index('index-id')
>>> document = index.document('document-id')
>>> field = document.field('field-name')
>>> field.add_value('britney spears', tokenization='atom')
>>> field.add_value('<h1>Britney Spears</h1>', tokenization='html')
```

38.5 Searching

After populating an index with documents, search through them by issuing a search query:

```
>>> from gcloud import search
>>> client = search.Client()
>>> index = client.index('index-id')
>>> query = client.query('britney spears')
>>> matching_documents = index.search(query) # API call
>>> for document in matching_documents:
...     print(document.id)
['document-id']
```

By default, all queries are sorted by the rank value set when the document was created. See: https://cloud.google.com/search/reference/rest/v1/projects/indexes/documents#resource_representation.google.cloudsearch.v1.Document

To sort differently, use the `order_by` parameter:

```
>>> ordered = client.query('britney spears', order_by=['field1', '-field2'])
```

Note that the `-` character before `field2` means that this query will be sorted ascending by `field1` and then descending by `field2`.

To limit the fields to be returned in the match, use the `fields` parameter:

```
>>> projected = client.query('britney spears', fields=['field1', 'field2'])
```

Search Client

Client for interacting with the Google Cloud search API.

class `gcloud.search.client.Client` (*project=None, credentials=None, http=None*)
 Bases: `gcloud.client.JSONClient`

Client to bundle configuration needed for API requests.

Parameters

- **project** (*string*) – the project which the client acts on behalf of. Will be passed when creating an index. If not passed, falls back to the default inferred from the environment.
- **credentials** (`oauth2client.client.OAuth2Credentials` or `NoneType`) – The OAuth2 Credentials to use for the connection owned by this client. If not passed (and if no `http` object is passed), falls back to the default inferred from the environment.
- **http** (`httplib2.Http` or class that defines `request()`) – An optional HTTP object to make requests. If not passed, an `http` object is created that is bound to the `credentials` for the current object.

index (*name*)

Construct an index bound to this client.

Parameters **name** (*string*) – Name of the index.

Return type `gcloud.search.index.Index`

Returns a new `Index` instance

list_indexes (*max_results=None, page_token=None, view=None, prefix=None*)

List indexes for the project associated with this client.

See: <https://cloud.google.com/search/reference/rest/v1/indexes/list>

Parameters

- **max_results** (*int*) – maximum number of indexes to return, If not passed, defaults to a value set by the API.
- **page_token** (*string*) – opaque marker for the next “page” of indexes. If not passed, the API will return the first page of indexes.
- **view** (*string*) – One of ‘ID_ONLY’ (return only the index ID; the default) or ‘FULL’ (return information on indexed fields).
- **prefix** (*string*) – return only indexes whose ID starts with `prefix`.

Return type tuple, (list, str)

Returns list of `gcloud.dns.index.Index`, plus a “next page token” string: if the token is not `None`, indicates that more indexes can be retrieved with another call (pass that value as `page_token`).

39.1 Connection

Create / interact with gcloud search connections.

class `gcloud.search.connection.Connection` (*credentials=None, http=None*)

Bases: `gcloud.connection.JSONConnection`

A connection to Google Cloud Search via the JSON REST API.

API_BASE_URL = `'https://cloudsearch.googleapis.com'`

The base of the API call URL.

API_URL_TEMPLATE = `'{api_base_url}/{api_version}{path}'`

A template for the URL of a particular API call.

API_VERSION = `'v1'`

The version of the API, used in building the API call's URL.

SCOPE = `('https://www.googleapis.com/auth/cloudsearch',)`

The scopes required for authenticating as a Cloud Search consumer.

Define API Indexes.

class `gcloud.search.index.Index` (*name*, *client*)

Bases: `object`

Indexes are containers for documents.

See: <https://cloud.google.com/search/reference/rest/v1/indexes>

Parameters

- **name** (*string*) – the name of the index
- **client** (`gcloud.dns.client.Client`) – A client which holds credentials and project configuration for the index (which requires a project).

atom_fields

Names of atom fields in the index.

Return type list of string, or None

Returns names of atom fields in the index, or None if no resource information is available.

date_fields

Names of date fields in the index.

Return type list of string, or None

Returns names of date fields in the index, or None if no resource information is available.

document (*name*, *rank=None*)

Construct a document bound to this index.

Parameters

- **name** (*string*) – Name of the document.
- **rank** (*integer*) – Rank of the document (defaults to a server-assigned value based on timestamp).

Return type `gcloud.search.document.Document`

Returns a new `Document` instance

classmethod `from_api_repr` (*resource*, *client*)

Factory: construct an index given its API representation

Parameters

- **resource** (*dict*) – index resource representation returned from the API

- **client** (*gcloud.dns.client.Client*) – Client which holds credentials and project configuration for the index.

Return type `gcloud.dns.index.Index`

Returns Index parsed from resource.

geo_fields

Names of geo fields in the index.

Return type list of string, or None

Returns names of geo fields in the index, or None if no resource information is available.

html_fields

Names of html fields in the index.

Return type list of string, or None

Returns names of html fields in the index, or None if no resource information is available.

list_documents (*max_results=None, page_token=None, view=None*)

List documents created within this index.

See: <https://cloud.google.com/search/reference/rest/v1/projects/indexes/documents/list>

Parameters

- **max_results** (*int*) – maximum number of indexes to return, If not passed, defaults to a value set by the API.
- **page_token** (*string*) – opaque marker for the next “page” of indexes. If not passed, the API will return the first page of indexes.
- **view** (*string*) – One of ‘ID_ONLY’ (return only the document ID; the default) or ‘FULL’ (return the full resource representation for the document, including field values)

Return type tuple, (list, str)

Returns list of `gcloud.dns.document.Document`, plus a “next page token” string: if the token is not None, indicates that more indexes can be retrieved with another call (pass that value as `page_token`).

number_fields

Names of number fields in the index.

Return type list of string, or None

Returns names of number fields in the index, or None if no resource information is available.

path

URL path for the index’s APIs.

Return type `string`

Returns the path based on project and dataste name.

project

Project bound to the index.

Return type `string`

Returns the project (derived from the client).

search (*query, max_results=None, page_token=None, field_expressions=None, order_by=None, matched_count_accuracy=None, scorer=None, scorer_size=None, return_fields=None*)

Search documents created within this index.

See: <https://cloud.google.com/search/reference/rest/v1/projects/indexes/search>

Parameters

- **query** (*string*) – query string (see <https://cloud.google.com/search/query>).
- **max_results** (*int*) – maximum number of indexes to return, If not passed, defaults to a value set by the API.
- **page_token** (*string*) – opaque marker for the next “page” of indexes. If not passed, the API will return the first page of indexes.
- **field_expressions** (dict, or `NoneType`) – mapping of field name -> expression for use in ‘order_by’ or ‘return_fields’
- **order_by** (sequence of string, or `NoneType`) – list of field names (plus optional ‘ desc’ suffix) specifying ordering of results.
- **matched_count_accuracy** (integer or `NoneType`) – minimum accuracy for matched count returned
- **return_fields** (sequence of string, or `NoneType`) – list of field names to be returned.
- **scorer** (string or `NoneType`) – name of scorer function (e.g., “generic”).
- **scorer_size** (integer or `NoneType`) – max number of top results pass to scorer function.

Return type tuple, (list, str, int)

Returns list of `gcloud.dns.document.Document`, plus a “next page token” string, and a “matched count”. If the token is not `None`, indicates that more indexes can be retrieved with another call (pass that value as `page_token`). The “matched count” indicates the total number of documents matching the query string.

text_fields

Names of text fields in the index.

Return type list of string, or `None`

Returns names of text fields in the index, or `None` if no resource information is available.

Documents

Define API Document.

class `gcloud.search.document.Document` (*name*, *index*, *rank=None*)
Bases: `object`

Documents hold values for search within indexes.

See: <https://cloud.google.com/search/reference/rest/v1/projects/indexes/documents>

Parameters

- **name** (*string*) – the name of the document
- **index** (`gcloud.search.index.Index`) – the index to which the document belongs.
- **rank** (*positive integer*) – override the server-generated rank for ordering the document within in queries. If not passed, the server generates a timestamp-based value. See the `rank` entry on the page above for details.

create (*client=None*)

API call: create the document via a PUT request

See: <https://cloud.google.com/search/reference/rest/v1/projects/indexes/documents/create>

Parameters **client** (`gcloud.search.client.Client` or `NoneType`) – the client to use. If not passed, falls back to the `client` stored on the current document's index.

delete (*client=None*)

API call: delete the document via a DELETE request.

See: <https://cloud.google.com/search/reference/rest/v1/projects/indexes/documents/delete>

Parameters **client** (`gcloud.search.client.Client` or `NoneType`) – the client to use. If not passed, falls back to the `client` stored on the current document's index.

exists (*client=None*)

API call: test existence of the document via a GET request

See <https://cloud.google.com/search/reference/rest/v1/projects/indexes/documents/get>

Parameters **client** (`gcloud.search.client.Client` or `NoneType`) – the client to use. If not passed, falls back to the `client` stored on the current document's index.

field (*name*)

Construct a Field instance.

Parameters **name** (*string*) – field's name

classmethod `from_api_repr` (*resource*, *index*)

Factory: construct a document given its API representation

Parameters

- **resource** (*dict*) – document resource representation returned from the API
- **index** (*gcloud.search.index.Index*) – Index holding the document.

Return type *gcloud.search.document.Document*

Returns Document parsed from *resource*.

path

URL path for the document's APIs

reload (*client=None*)

API call: sync local document configuration via a GET request

See <https://cloud.google.com/search/reference/rest/v1/projects/indexes/documents/get>

Parameters **client** (*gcloud.search.client.Client* or *NoneType*) – the client to use. If not passed, falls back to the `client` stored on the current document's index.

class `gcloud.search.document.Field` (*name*)

Bases: `object`

Fields hold values for a given document

See: <https://cloud.google.com/search/reference/rest/google/cloudsearch/v1/FieldValueList>

Parameters **name** (*string*) – field name

add_value (*value*, ***kw*)

Add a value to the field.

Selects type of value instance based on type of *value*.

Parameters

- **value** (*string*, *integer*, *float*, *datetime*, or *tuple (float, float)*) – the field value to add.
- **kw** – extra keyword arguments to be passed to the value instance constructor. Currently, only *StringValue* expects / honors additional parameters.

Raises `ValueError` if unable to match the type of *value*.

class `gcloud.search.document.GeoValue` (*geo_value*)

Bases: `object`

GeoValues hold individual latitude/longitude values for a given field See: <https://cloud.google.com/search/reference/rest/google/cloudsearch/v1/FieldValue>

Parameters **geo_value** (*tuple, (float, float)*) – latitude, longitude

value_type = 'geo'

class `gcloud.search.document.NumberValue` (*number_value*)

Bases: `object`

NumberValues hold individual numeric values for a given field

See: <https://cloud.google.com/search/reference/rest/google/cloudsearch/v1/FieldValue>

Parameters **number_value** (*integer, float (long on Python2)*) – the actual value.

value_type = 'number'

class `gcloud.search.document.StringValue` (*string_value*, *string_format=None*, *language=None*)

Bases: `object`

StringValue holds individual text values for a given field

See: <https://cloud.google.com/search/reference/rest/google/cloudsearch/v1/FieldValue>

Parameters

- **string_value** (*string*) – the actual value.
- **string_format** (*string*) – how the value should be indexed: one of ‘ATOM’, ‘TEXT’, ‘HTML’ (leave as `None` to use the server-supplied default).
- **language** (*string*) – Human language of the text. Should be an ISO 639-1 language code.

value_type = ‘string’

class `gcloud.search.document.TimestampValue` (*timestamp_value*)

Bases: `object`

TimestampValues hold individual datetime values for a given field See: <https://cloud.google.com/search/reference/rest/google/cloudsearch/v1/FieldValue>

Parameters **timestamp_value** (`class:datetime.datetime`) – the actual value.

value_type = ‘timestamp’

Getting started

The `gcloud` library is `pip` install-able:

```
$ pip install gcloud
```

If you have trouble installing `pycrypto` or `pyopenssl` (and you're on Ubuntu), you can try install the precompiled packages:

```
$ sudo apt-get install python-crypto python-openssl
```

If you want to install everything with `pip`, try installing the dev packages beforehand:

```
$ sudo apt-get install python-dev libssl-dev
```

If you want to install `gcloud-python` from source, you can clone the repository from GitHub:

```
$ git clone git://github.com/GoogleCloudPlatform/gcloud-python.git
$ cd gcloud-python
$ python setup.py install
```

42.1 Cloud Datastore

Google Cloud Datastore is a fully managed, schemaless database for storing non-relational data.

```
from gcloud import datastore

client = datastore.Client()
key = client.key('Person')

entity = datastore.Entity(key=key)
entity['name'] = 'Your name'
entity['age'] = 25
client.put(entity)
```

42.2 Cloud Storage

Google Cloud Storage allows you to store data on Google infrastructure.

```
from gcloud import storage

client = storage.Client()
bucket = client.get_bucket('<your-bucket-name>')
blob = bucket.blob('my-test-file.txt')
blob.upload_from_string('this is test content!')
```


g

`gcloud.bigquery.client`, 107
`gcloud.bigquery.connection`, 109
`gcloud.bigquery.dataset`, 111
`gcloud.bigquery.job`, 115
`gcloud.bigquery.query`, 127
`gcloud.bigquery.table`, 121
`gcloud.client`, 1
`gcloud.connection`, 9
`gcloud.credentials`, 5
`gcloud.datastore.batch`, 45
`gcloud.datastore.client`, 25
`gcloud.datastore.connection`, 27
`gcloud.datastore.entity`, 31
`gcloud.datastore.helpers`, 49
`gcloud.datastore.key`, 33
`gcloud.datastore.query`, 37
`gcloud.datastore.transaction`, 41
`gcloud.dns.changes`, 153
`gcloud.dns.client`, 145
`gcloud.dns.connection`, 146
`gcloud.dns.resource_record_set`, 151
`gcloud.dns.zone`, 147
`gcloud.environment_vars`, 17
`gcloud.exceptions`, 13
`gcloud.pubsub.client`, 83
`gcloud.pubsub.connection`, 84
`gcloud.pubsub.message`, 95
`gcloud.pubsub.subscription`, 91
`gcloud.pubsub.topic`, 87
`gcloud.resource_manager.client`, 133
`gcloud.resource_manager.connection`, 134
`gcloud.resource_manager.project`, 137
`gcloud.search.client`, 159
`gcloud.search.connection`, 160
`gcloud.search.document`, 165
`gcloud.search.index`, 161
`gcloud.storage.acl`, 71
`gcloud.storage.batch`, 77
`gcloud.storage.blob`, 55
`gcloud.storage.bucket`, 63
`gcloud.storage.client`, 51
`gcloud.storage.connection`, 53

A

- access_grants (gcloud.bigquery.dataset.Dataset attribute), 112
- AccessGrant (class in gcloud.bigquery.dataset), 111
- acknowledge() (gcloud.pubsub.subscription.Subscription method), 91
- ACL (class in gcloud.storage.acl), 72
- acl (gcloud.storage.blob.Blob attribute), 55
- acl (gcloud.storage.bucket.Bucket attribute), 63
- add_entity() (gcloud.storage.acl.ACL method), 72
- add_filter() (gcloud.datastore.query.Query method), 38
- add_record_set() (gcloud.dns.changes.Changes method), 153
- add_value() (gcloud.search.document.Field method), 166
- additions (gcloud.dns.changes.Changes attribute), 153
- all() (gcloud.storage.acl.ACL method), 72
- all_authenticated() (gcloud.storage.acl.ACL method), 72
- allocate_ids() (gcloud.datastore.client.Client method), 25
- allocate_ids() (gcloud.datastore.connection.Connection method), 28
- allow_jagged_rows (gcloud.bigquery.job.LoadTableFromStorageJob attribute), 117
- allow_large_results (gcloud.bigquery.job.QueryJob attribute), 118
- allow_quoted_newlines (gcloud.bigquery.job.LoadTableFromStorageJob attribute), 117
- ALLOWED (gcloud.bigquery.job.Compression attribute), 115
- ALLOWED (gcloud.bigquery.job.CreateDisposition attribute), 116
- ALLOWED (gcloud.bigquery.job.DestinationFormat attribute), 116
- ALLOWED (gcloud.bigquery.job.Encoding attribute), 116
- ALLOWED (gcloud.bigquery.job.QueryPriority attribute), 119
- ALLOWED (gcloud.bigquery.job.SourceFormat attribute), 119
- ALLOWED (gcloud.bigquery.job.WriteDisposition attribute), 119
- ancestor (gcloud.datastore.query.Query attribute), 38
- API_BASE_URL (gcloud.bigquery.connection.Connection attribute), 109
- API_BASE_URL (gcloud.connection.JSONConnection attribute), 10
- API_BASE_URL (gcloud.datastore.connection.Connection attribute), 27
- API_BASE_URL (gcloud.dns.connection.Connection attribute), 146
- API_BASE_URL (gcloud.pubsub.connection.Connection attribute), 84
- API_BASE_URL (gcloud.resource_manager.connection.Connection attribute), 135
- API_BASE_URL (gcloud.search.connection.Connection attribute), 160
- API_BASE_URL (gcloud.storage.connection.Connection attribute), 53
- API_BASE_URL (in module gcloud.connection), 9
- api_request() (gcloud.connection.JSONConnection method), 10
- API_URL_TEMPLATE (gcloud.bigquery.connection.Connection attribute), 110
- API_URL_TEMPLATE (gcloud.connection.JSONConnection attribute), 10
- API_URL_TEMPLATE (gcloud.datastore.connection.Connection attribute), 28
- API_URL_TEMPLATE (gcloud.dns.connection.Connection attribute), 146
- API_URL_TEMPLATE (gcloud.pubsub.connection.Connection attribute), 84
- API_URL_TEMPLATE (gcloud.resource_manager.connection.Connection attribute), 135
- API_URL_TEMPLATE (gcloud.search.connection.Connection attribute), 160
- API_URL_TEMPLATE (gcloud.storage.connection.Connection attribute), 53
- API_VERSION (gcloud.bigquery.connection.Connection attribute), 110
- API_VERSION (gcloud.connection.JSONConnection attribute), 10
- API_VERSION (gcloud.datastore.connection.Connection

attribute), 28
 API_VERSION (gcloud.dns.connection.Connection attribute), 146
 API_VERSION (gcloud.pubsub.connection.Connection attribute), 84
 API_VERSION (gcloud.resource_manager.connection.Connection attribute), 135
 API_VERSION (gcloud.search.connection.Connection attribute), 160
 API_VERSION (gcloud.storage.connection.Connection attribute), 53
 atom_fields (gcloud.search.index.Index attribute), 161
 attributes (gcloud.pubsub.message.Message attribute), 95
 AVRO (gcloud.bigquery.job.DestinationFormat attribute), 116

B

BadRequest, 13
 Batch (class in gcloud.datastore.batch), 45
 Batch (class in gcloud.pubsub.topic), 87
 Batch (class in gcloud.storage.batch), 77
 BATCH (gcloud.bigquery.job.QueryPriority attribute), 119
 batch() (gcloud.datastore.client.Client method), 25
 batch() (gcloud.pubsub.topic.Topic method), 87
 batch() (gcloud.storage.client.Client method), 51
 begin() (gcloud.datastore.batch.Batch method), 45
 begin() (gcloud.datastore.transaction.Transaction method), 42
 begin_transaction() (gcloud.datastore.connection.Connection method), 28
 Blob (class in gcloud.storage.blob), 55
 blob() (gcloud.storage.bucket.Bucket method), 63
 Bucket (class in gcloud.storage.bucket), 63
 bucket() (gcloud.storage.client.Client method), 51
 BucketACL (class in gcloud.storage.acl), 74
 build_api_url() (gcloud.connection.JSONConnection class method), 11
 build_api_url() (gcloud.datastore.connection.Connection method), 28
 build_api_url() (gcloud.pubsub.connection.Connection method), 84

C

cache_control (gcloud.storage.blob.Blob attribute), 55
 cache_hit (gcloud.bigquery.query.QueryResults attribute), 127
 Changes (class in gcloud.dns.changes), 153
 changes() (gcloud.dns.zone.ManagedZone method), 147
 chunk_size (gcloud.storage.blob.Blob attribute), 55
 clear() (gcloud.storage.acl.ACL method), 72
 Client (class in gcloud.bigquery.client), 107
 Client (class in gcloud.client), 1
 Client (class in gcloud.datastore.client), 25

Client (class in gcloud.dns.client), 145
 Client (class in gcloud.pubsub.client), 83
 Client (class in gcloud.resource_manager.client), 133
 Client (class in gcloud.search.client), 159
 Client (class in gcloud.storage.client), 51
 client (gcloud.storage.acl.ACL attribute), 72
 client (gcloud.storage.acl.BucketACL attribute), 74
 client (gcloud.storage.acl.ObjectACL attribute), 74
 client (gcloud.storage.blob.Blob attribute), 55
 client (gcloud.storage.bucket.Bucket attribute), 63
 ClientError, 13
 code (gcloud.exceptions.BadRequest attribute), 13
 code (gcloud.exceptions.Conflict attribute), 13
 code (gcloud.exceptions.Forbidden attribute), 13
 code (gcloud.exceptions.GCloudError attribute), 13
 code (gcloud.exceptions.InternalServerError attribute), 14
 code (gcloud.exceptions.LengthRequired attribute), 14
 code (gcloud.exceptions.MethodNotAllowed attribute), 14
 code (gcloud.exceptions.MovedPermanently attribute), 14
 code (gcloud.exceptions.NotFound attribute), 14
 code (gcloud.exceptions.NotImplemented attribute), 14
 code (gcloud.exceptions.NotModified attribute), 14
 code (gcloud.exceptions.PreconditionFailed attribute), 14
 code (gcloud.exceptions.RequestRangeNotSatisfiable attribute), 14
 code (gcloud.exceptions.ResumeIncomplete attribute), 15
 code (gcloud.exceptions.ServiceUnavailable attribute), 15
 code (gcloud.exceptions.TemporaryRedirect attribute), 15
 code (gcloud.exceptions.TooManyRequests attribute), 15
 code (gcloud.exceptions.Unauthorized attribute), 15
 commit() (gcloud.datastore.batch.Batch method), 46
 commit() (gcloud.datastore.connection.Connection method), 28
 commit() (gcloud.datastore.transaction.Transaction method), 42
 commit() (gcloud.pubsub.topic.Batch method), 87
 complete (gcloud.bigquery.query.QueryResults attribute), 127
 completed_key() (gcloud.datastore.key.Key method), 33
 component_count (gcloud.storage.blob.Blob attribute), 55
 Compression (class in gcloud.bigquery.job), 115
 compression (gcloud.bigquery.job.ExtractTableToStorageJob attribute), 116
 configure_website() (gcloud.storage.bucket.Bucket method), 63
 Conflict, 13
 Connection (class in gcloud.bigquery.connection), 109
 Connection (class in gcloud.connection), 9
 Connection (class in gcloud.datastore.connection), 27
 Connection (class in gcloud.dns.connection), 146

- Connection (class in gcloud.pubsub.connection), 84
- Connection (class in gcloud.resource_manager.connection), 134
- Connection (class in gcloud.search.connection), 160
- Connection (class in gcloud.storage.connection), 53
- connection (gcloud.datastore.batch.Batch attribute), 46
- connection (gcloud.datastore.transaction.Transaction attribute), 42
- connection (gcloud.storage.client.Client attribute), 51
- content_disposition (gcloud.storage.blob.Blob attribute), 55
- content_encoding (gcloud.storage.blob.Blob attribute), 56
- content_language (gcloud.storage.blob.Blob attribute), 56
- content_type (gcloud.storage.blob.Blob attribute), 56
- copy_blob() (gcloud.storage.bucket.Bucket method), 64
- copy_table() (gcloud.bigquery.client.Client method), 107
- CopyJob (class in gcloud.bigquery.job), 115
- cors (gcloud.storage.bucket.Bucket attribute), 64
- crc32c (gcloud.storage.blob.Blob attribute), 56
- create() (gcloud.bigquery.dataset.Dataset method), 112
- create() (gcloud.bigquery.table.Table method), 121
- create() (gcloud.dns.changes.Changes method), 153
- create() (gcloud.dns.zone.ManagedZone method), 147
- create() (gcloud.pubsub.subscription.Subscription method), 91
- create() (gcloud.pubsub.topic.Topic method), 88
- create() (gcloud.resource_manager.project.Project method), 137
- create() (gcloud.search.document.Document method), 165
- create() (gcloud.storage.bucket.Bucket method), 64
- create_bucket() (gcloud.storage.client.Client method), 52
- create_disposition (gcloud.bigquery.job.CopyJob attribute), 115
- create_disposition (gcloud.bigquery.job.LoadTableFromStorageJob attribute), 117
- create_disposition (gcloud.bigquery.job.QueryJob attribute), 118
- CREATE_IF_NEEDED (gcloud.bigquery.job.CreateDisposition attribute), 116
- CREATE_NEVER (gcloud.bigquery.job.CreateDisposition attribute), 116
- created (gcloud.bigquery.dataset.Dataset attribute), 112
- created (gcloud.bigquery.table.Table attribute), 121
- created (gcloud.dns.zone.ManagedZone attribute), 147
- CreateDisposition (class in gcloud.bigquery.job), 116
- credentials (gcloud.connection.Connection attribute), 9
- CREDENTIALS (in module gcloud.environment_vars), 17
- CSV (gcloud.bigquery.job.DestinationFormat attribute), 116
- CSV (gcloud.bigquery.job.SourceFormat attribute), 119
- current() (gcloud.datastore.batch.Batch method), 46
- current() (gcloud.datastore.transaction.Transaction method), 42
- current() (gcloud.storage.batch.Batch method), 77
- current_batch (gcloud.datastore.client.Client attribute), 25
- current_batch (gcloud.storage.client.Client attribute), 52
- current_transaction (gcloud.datastore.client.Client attribute), 25
- ## D
- Dataset (class in gcloud.bigquery.dataset), 111
- DATASET (in module gcloud.environment_vars), 17
- dataset() (gcloud.bigquery.client.Client method), 107
- dataset_id (gcloud.bigquery.dataset.Dataset attribute), 112
- dataset_name (gcloud.bigquery.table.Table attribute), 122
- DATASTORE_BACKUP (gcloud.bigquery.job.SourceFormat attribute), 119
- date_fields (gcloud.search.index.Index attribute), 161
- default_dataset (gcloud.bigquery.job.QueryJob attribute), 119
- default_dataset (gcloud.bigquery.query.QueryResults attribute), 127
- default_object_acl (gcloud.storage.bucket.Bucket attribute), 64
- default_table_expiration_ms (gcloud.bigquery.dataset.Dataset attribute), 112
- DefaultObjectACL (class in gcloud.storage.acl), 74
- delete() (gcloud.bigquery.dataset.Dataset method), 112
- delete() (gcloud.bigquery.table.Table method), 122
- delete() (gcloud.datastore.batch.Batch method), 46
- delete() (gcloud.datastore.client.Client method), 26
- delete() (gcloud.datastore.transaction.Transaction method), 43
- delete() (gcloud.dns.zone.ManagedZone method), 147
- delete() (gcloud.pubsub.subscription.Subscription method), 91
- delete() (gcloud.pubsub.topic.Topic method), 88
- delete() (gcloud.resource_manager.project.Project method), 137
- delete() (gcloud.search.document.Document method), 165
- delete() (gcloud.storage.blob.Blob method), 56
- delete() (gcloud.storage.bucket.Bucket method), 65
- delete_blob() (gcloud.storage.bucket.Bucket method), 65
- delete_blobs() (gcloud.storage.bucket.Bucket method), 65
- delete_multi() (gcloud.datastore.client.Client method), 26
- delete_record_set() (gcloud.dns.changes.Changes method), 153
- deletions (gcloud.dns.changes.Changes attribute), 153

- description (gcloud.bigquery.dataset.Dataset attribute), 112
- description (gcloud.bigquery.table.Table attribute), 122
- description (gcloud.dns.zone.ManagedZone attribute), 147
- destination (gcloud.bigquery.job.QueryJob attribute), 119
- destination_format (gcloud.bigquery.job.ExtractTableToStorageJob attribute), 116
- DestinationFormat (class in gcloud.bigquery.job), 116
- disable_logging() (gcloud.storage.bucket.Bucket method), 66
- disable_website() (gcloud.storage.bucket.Bucket method), 66
- Document (class in gcloud.search.document), 165
- document() (gcloud.search.index.Index method), 161
- domain() (gcloud.storage.acl.ACL method), 72
- download_as_string() (gcloud.storage.blob.Blob method), 56
- download_to_file() (gcloud.storage.blob.Blob method), 57
- download_to_filename() (gcloud.storage.blob.Blob method), 57
- ## E
- enable_logging() (gcloud.storage.bucket.Bucket method), 66
- Encoding (class in gcloud.bigquery.job), 116
- encoding (gcloud.bigquery.job.LoadTableFromStorageJob attribute), 117
- Entity (class in gcloud.datastore.entity), 31
- entity() (gcloud.storage.acl.ACL method), 72
- entity_from_dict() (gcloud.storage.acl.ACL method), 73
- entity_from_protobuf() (in module gcloud.datastore.helpers), 49
- ENTITY_TYPES (gcloud.bigquery.dataset.AccessGrant attribute), 111
- environment variable
- GOOGLE_APPLICATION_CREDENTIALS, 6
- errors (gcloud.bigquery.query.QueryResults attribute), 127
- errors (gcloud.exceptions.GCloudError attribute), 13
- etag (gcloud.bigquery.dataset.Dataset attribute), 112
- etag (gcloud.bigquery.table.Table attribute), 122
- etag (gcloud.storage.blob.Blob attribute), 57
- etag (gcloud.storage.bucket.Bucket attribute), 66
- exclude_from_indexes (gcloud.datastore.entity.Entity attribute), 32
- exists() (gcloud.bigquery.dataset.Dataset method), 112
- exists() (gcloud.bigquery.table.Table method), 122
- exists() (gcloud.dns.changes.Changes method), 154
- exists() (gcloud.dns.zone.ManagedZone method), 148
- exists() (gcloud.pubsub.subscription.Subscription method), 92
- exists() (gcloud.pubsub.topic.Topic method), 88
- exists() (gcloud.resource_manager.project.Project method), 138
- exists() (gcloud.search.document.Document method), 165
- exists() (gcloud.storage.blob.Blob method), 57
- exists() (gcloud.storage.bucket.Bucket method), 66
- exists() (gcloud.bigquery.table.Table attribute), 122
- extract_table_to_storage() (gcloud.bigquery.client.Client method), 107
- ExtractTableToStorageJob (class in gcloud.bigquery.job), 116
- ## F
- fetch() (gcloud.datastore.query.Query method), 38
- fetch_data() (gcloud.bigquery.query.QueryResults method), 127
- fetch_data() (gcloud.bigquery.table.Table method), 122
- fetch_project() (gcloud.resource_manager.client.Client method), 133
- Field (class in gcloud.search.document), 166
- field() (gcloud.search.document.Document method), 165
- field_delimiter (gcloud.bigquery.job.ExtractTableToStorageJob attribute), 116
- field_delimiter (gcloud.bigquery.job.LoadTableFromStorageJob attribute), 117
- filters (gcloud.datastore.query.Query attribute), 39
- finish() (gcloud.storage.batch.Batch method), 77
- flat_path (gcloud.datastore.key.Key attribute), 34
- flatten_results (gcloud.bigquery.job.QueryJob attribute), 119
- Forbidden, 13
- friendly_name (gcloud.bigquery.dataset.Dataset attribute), 112
- friendly_name (gcloud.bigquery.table.Table attribute), 123
- from_api_repr() (gcloud.bigquery.dataset.Dataset class method), 113
- from_api_repr() (gcloud.bigquery.job.CopyJob class method), 115
- from_api_repr() (gcloud.bigquery.job.ExtractTableToStorageJob class method), 116
- from_api_repr() (gcloud.bigquery.job.LoadTableFromStorageJob class method), 117
- from_api_repr() (gcloud.bigquery.job.QueryJob class method), 119
- from_api_repr() (gcloud.bigquery.table.Table class method), 123
- from_api_repr() (gcloud.dns.changes.Changes class method), 154
- from_api_repr() (gcloud.dns.resource_record_set.ResourceRecordSet class method), 151
- from_api_repr() (gcloud.dns.zone.ManagedZone class method), 148

- [from_api_repr\(\) \(gcloud.pubsub.message.Message class method\), 95](#)
[from_api_repr\(\) \(gcloud.pubsub.subscription.Subscription class method\), 92](#)
[from_api_repr\(\) \(gcloud.pubsub.topic.Topic class method\), 88](#)
[from_api_repr\(\) \(gcloud.resource_manager.project.Project class method\), 138](#)
[from_api_repr\(\) \(gcloud.search.document.Document class method\), 165](#)
[from_api_repr\(\) \(gcloud.search.index.Index class method\), 161](#)
[from_service_account_json\(\) \(gcloud.client.Client method\), 1](#)
[from_service_account_json\(\) \(gcloud.client.JSONClient method\), 2](#)
[from_service_account_p12\(\) \(gcloud.client.Client method\), 1](#)
[from_service_account_p12\(\) \(gcloud.client.JSONClient method\), 2](#)
[full_name \(gcloud.pubsub.topic.Topic attribute\), 88](#)
[full_name \(gcloud.resource_manager.project.Project attribute\), 138](#)
- ## G
- [GCD_DATASET \(in module gcloud.environment_vars\), 17](#)
[GCD_HOST \(in module gcloud.environment_vars\), 17](#)
[gcloud.bigquery.client \(module\), 107](#)
[gcloud.bigquery.connection \(module\), 109](#)
[gcloud.bigquery.dataset \(module\), 111](#)
[gcloud.bigquery.job \(module\), 115](#)
[gcloud.bigquery.query \(module\), 127](#)
[gcloud.bigquery.table \(module\), 121](#)
[gcloud.client \(module\), 1](#)
[gcloud.connection \(module\), 9](#)
[gcloud.credentials \(module\), 5](#)
[gcloud.datastore.batch \(module\), 45](#)
[gcloud.datastore.client \(module\), 25](#)
[gcloud.datastore.connection \(module\), 27](#)
[gcloud.datastore.entity \(module\), 31](#)
[gcloud.datastore.helpers \(module\), 49](#)
[gcloud.datastore.key \(module\), 33](#)
[gcloud.datastore.query \(module\), 37](#)
[gcloud.datastore.transaction \(module\), 41](#)
[gcloud.dns.changes \(module\), 153](#)
[gcloud.dns.client \(module\), 145](#)
[gcloud.dns.connection \(module\), 146](#)
[gcloud.dns.resource_record_set \(module\), 151](#)
[gcloud.dns.zone \(module\), 147](#)
[gcloud.environment_vars \(module\), 17](#)
[gcloud.exceptions \(module\), 13](#)
[gcloud.pubsub.client \(module\), 83](#)
[gcloud.pubsub.connection \(module\), 84](#)
[gcloud.pubsub.message \(module\), 95](#)
[gcloud.pubsub.subscription \(module\), 91](#)
[gcloud.pubsub.topic \(module\), 87](#)
[gcloud.resource_manager.client \(module\), 133](#)
[gcloud.resource_manager.connection \(module\), 134](#)
[gcloud.resource_manager.project \(module\), 137](#)
[gcloud.search.client \(module\), 159](#)
[gcloud.search.connection \(module\), 160](#)
[gcloud.search.document \(module\), 165](#)
[gcloud.search.index \(module\), 161](#)
[gcloud.storage.acl \(module\), 71](#)
[gcloud.storage.batch \(module\), 77](#)
[gcloud.storage.blob \(module\), 55](#)
[gcloud.storage.bucket \(module\), 63](#)
[gcloud.storage.client \(module\), 51](#)
[gcloud.storage.connection \(module\), 53](#)
[GCloudError, 13](#)
[generate_signed_url\(\) \(gcloud.storage.blob.Blob method\), 57](#)
[generate_signed_url\(\) \(in module gcloud.credentials\), 5](#)
[generation \(gcloud.storage.blob.Blob attribute\), 58](#)
[geo_fields \(gcloud.search.index.Index attribute\), 162](#)
[GeoValue \(class in gcloud.search.document\), 166](#)
[get\(\) \(gcloud.datastore.client.Client method\), 26](#)
[get_blob\(\) \(gcloud.storage.bucket.Bucket method\), 66](#)
[get_bucket\(\) \(gcloud.storage.client.Client method\), 52](#)
[get_credentials\(\) \(in module gcloud.credentials\), 6](#)
[get_entities\(\) \(gcloud.storage.acl.ACL method\), 73](#)
[get_entity\(\) \(gcloud.storage.acl.ACL method\), 73](#)
[get_for_service_account_json\(\) \(in module gcloud.credentials\), 6](#)
[get_for_service_account_p12\(\) \(in module gcloud.credentials\), 7](#)
[get_logging\(\) \(gcloud.storage.bucket.Bucket method\), 67](#)
[get_multi\(\) \(gcloud.datastore.client.Client method\), 26](#)
[GOOGLE_APPLICATION_CREDENTIALS, 6](#)
[group\(\) \(gcloud.storage.acl.ACL method\), 73](#)
[group_by \(gcloud.datastore.query.Query attribute\), 39](#)
[GZIP \(gcloud.bigquery.job.Compression attribute\), 115](#)
- ## H
- [has_entity\(\) \(gcloud.storage.acl.ACL method\), 73](#)
[html_fields \(gcloud.search.index.Index attribute\), 162](#)
[http \(gcloud.connection.Connection attribute\), 10](#)
- ## I
- [id \(gcloud.datastore.key.Key attribute\), 34](#)
[id \(gcloud.datastore.transaction.Transaction attribute\), 43](#)
[id \(gcloud.storage.blob.Blob attribute\), 58](#)
[id \(gcloud.storage.bucket.Bucket attribute\), 67](#)
[id_or_name \(gcloud.datastore.key.Key attribute\), 34](#)
[ignore_unknown_values \(gcloud.bigquery.job.LoadTableFromStorageJob attribute\), 117](#)
[Index \(class in gcloud.search.index\), 161](#)

- [index\(\)](#) (gcloud.search.client.Client method), 159
[input_file_bytes](#) (gcloud.bigquery.job.LoadTableFromStorageJob attribute), 117
[input_files](#) (gcloud.bigquery.job.LoadTableFromStorageJob attribute), 118
[insert_data\(\)](#) (gcloud.bigquery.table.Table method), 123
[INTERACTIVE](#) (gcloud.bigquery.job.QueryPriority attribute), 119
[InternalServerError](#), 13
[is_partial](#) (gcloud.datastore.key.Key attribute), 34
[ISO_8559_1](#) (gcloud.bigquery.job.Encoding attribute), 116
[Iterator](#) (class in gcloud.datastore.query), 37
- ## J
- [job](#) (gcloud.bigquery.query.QueryResults attribute), 128
[job_from_resource\(\)](#) (gcloud.bigquery.client.Client method), 108
[JSONClient](#) (class in gcloud.client), 2
[JSONConnection](#) (class in gcloud.connection), 10
- ## K
- [Key](#) (class in gcloud.datastore.key), 33
[key\(\)](#) (gcloud.datastore.client.Client method), 27
[key_filter\(\)](#) (gcloud.datastore.query.Query method), 39
[key_from_protobuf\(\)](#) (in module gcloud.datastore.helpers), 49
[keys_only\(\)](#) (gcloud.datastore.query.Query method), 39
[kind](#) (gcloud.datastore.entity.Entity attribute), 32
[kind](#) (gcloud.datastore.key.Key attribute), 34
[kind](#) (gcloud.datastore.query.Query attribute), 39
- ## L
- [LengthRequired](#), 14
[lifecycle_rules](#) (gcloud.storage.bucket.Bucket attribute), 67
[list_blobs\(\)](#) (gcloud.storage.bucket.Bucket method), 67
[list_buckets\(\)](#) (gcloud.storage.client.Client method), 52
[list_changes\(\)](#) (gcloud.dns.zone.ManagedZone method), 148
[list_datasets\(\)](#) (gcloud.bigquery.client.Client method), 108
[list_documents\(\)](#) (gcloud.search.index.Index method), 162
[list_indexes\(\)](#) (gcloud.search.client.Client method), 159
[list_jobs\(\)](#) (gcloud.bigquery.client.Client method), 108
[list_projects\(\)](#) (gcloud.resource_manager.client.Client method), 133
[list_resource_record_sets\(\)](#) (gcloud.dns.zone.ManagedZone method), 148
[list_subscriptions\(\)](#) (gcloud.pubsub.client.Client method), 83
[list_tables\(\)](#) (gcloud.bigquery.dataset.Dataset method), 113
[list_topics\(\)](#) (gcloud.pubsub.client.Client method), 83
[list_zones\(\)](#) (gcloud.dns.client.Client method), 145
[load_table_from_storage\(\)](#) (gcloud.bigquery.client.Client method), 109
[loaded](#) (gcloud.storage.acl.ACL attribute), 73
[LoadTableFromStorageJob](#) (class in gcloud.bigquery.job), 117
[location](#) (gcloud.bigquery.dataset.Dataset attribute), 113
[location](#) (gcloud.bigquery.table.Table attribute), 123
[location](#) (gcloud.storage.bucket.Bucket attribute), 68
[lookup\(\)](#) (gcloud.datastore.connection.Connection method), 29
[lookup_bucket\(\)](#) (gcloud.storage.client.Client method), 53
- ## M
- [make_exception\(\)](#) (in module gcloud.exceptions), 15
[make_public\(\)](#) (gcloud.storage.blob.Blob method), 58
[make_public\(\)](#) (gcloud.storage.bucket.Bucket method), 68
[ManagedZone](#) (class in gcloud.dns.zone), 147
[max_bad_records](#) (gcloud.bigquery.job.LoadTableFromStorageJob attribute), 118
[max_results](#) (gcloud.bigquery.query.QueryResults attribute), 128
[md5_hash](#) (gcloud.storage.blob.Blob attribute), 58
[media_link](#) (gcloud.storage.blob.Blob attribute), 58
[Message](#) (class in gcloud.pubsub.message), 95
[metadata](#) (gcloud.storage.blob.Blob attribute), 59
[metageneration](#) (gcloud.storage.blob.Blob attribute), 59
[metageneration](#) (gcloud.storage.bucket.Bucket attribute), 68
[MethodNotAllowed](#), 14
[MIMEApplicationHTTP](#) (class in gcloud.storage.batch), 77
[modified](#) (gcloud.bigquery.dataset.Dataset attribute), 113
[modified](#) (gcloud.bigquery.table.Table attribute), 123
[modify_ack_deadline\(\)](#) (gcloud.pubsub.subscription.Subscription method), 92
[modify_push_configuration\(\)](#) (gcloud.pubsub.subscription.Subscription method), 92
[MovedPermanently](#), 14
[mutations](#) (gcloud.datastore.batch.Batch attribute), 46
[mutations](#) (gcloud.datastore.transaction.Transaction attribute), 43
- ## N
- [name](#) (gcloud.bigquery.query.QueryResults attribute), 128
[name](#) (gcloud.datastore.key.Key attribute), 34
[name](#) (gcloud.dns.changes.Changes attribute), 154

- name_server_set (gcloud.dns.zone.ManagedZone attribute), 149
- name_servers (gcloud.dns.zone.ManagedZone attribute), 149
- namespace (gcloud.datastore.batch.Batch attribute), 46
- namespace (gcloud.datastore.key.Key attribute), 34
- namespace (gcloud.datastore.query.Query attribute), 39
- namespace (gcloud.datastore.transaction.Transaction attribute), 43
- new_project() (gcloud.resource_manager.client.Client method), 134
- NEWLINE_DELIMITED_JSON (gcloud.bigquery.job.DestinationFormat attribute), 116
- NEWLINE_DELIMITED_JSON (gcloud.bigquery.job.SourceFormat attribute), 119
- next_page() (gcloud.datastore.query.Iterator method), 37
- NoContent (class in gcloud.storage.batch), 77
- NONE (gcloud.bigquery.job.Compression attribute), 115
- NotFound, 14
- NotImplemented, 14
- NotModified, 14
- num_bytes (gcloud.bigquery.table.Table attribute), 124
- num_rows (gcloud.bigquery.table.Table attribute), 124
- number_fields (gcloud.search.index.Index attribute), 162
- NumberValue (class in gcloud.search.document), 166
- ## O
- ObjectACL (class in gcloud.storage.acl), 74
- OPERATORS (gcloud.datastore.query.Query attribute), 38
- order (gcloud.datastore.query.Query attribute), 39
- output_bytes (gcloud.bigquery.job.LoadTableFromStorageJob attribute), 118
- output_rows (gcloud.bigquery.job.LoadTableFromStorageJob attribute), 118
- owner (gcloud.storage.blob.Blob attribute), 59
- owner (gcloud.storage.bucket.Bucket attribute), 68
- ## P
- page_token (gcloud.bigquery.query.QueryResults attribute), 128
- parent (gcloud.datastore.key.Key attribute), 34
- patch() (gcloud.bigquery.dataset.Dataset method), 113
- patch() (gcloud.bigquery.table.Table method), 124
- path (gcloud.bigquery.dataset.Dataset attribute), 113
- path (gcloud.bigquery.table.Table attribute), 124
- path (gcloud.datastore.key.Key attribute), 34
- path (gcloud.dns.changes.Changes attribute), 154
- path (gcloud.dns.zone.ManagedZone attribute), 149
- path (gcloud.pubsub.subscription.Subscription attribute), 92
- path (gcloud.pubsub.topic.Topic attribute), 88
- path (gcloud.resource_manager.project.Project attribute), 138
- path (gcloud.search.document.Document attribute), 166
- path (gcloud.search.index.Index attribute), 162
- path (gcloud.storage.blob.Blob attribute), 59
- path (gcloud.storage.bucket.Bucket attribute), 68
- path_helper() (gcloud.storage.blob.Blob static method), 59
- path_helper() (gcloud.storage.bucket.Bucket static method), 68
- PreconditionFailed, 14
- preserve_nulls (gcloud.bigquery.query.QueryResults attribute), 128
- print_header (gcloud.bigquery.job.ExtractTableToStorageJob attribute), 117
- priority (gcloud.bigquery.job.QueryJob attribute), 119
- Project (class in gcloud.resource_manager.project), 137
- project (gcloud.bigquery.dataset.Dataset attribute), 114
- project (gcloud.bigquery.query.QueryResults attribute), 128
- project (gcloud.bigquery.table.Table attribute), 124
- project (gcloud.datastore.batch.Batch attribute), 46
- project (gcloud.datastore.key.Key attribute), 35
- project (gcloud.datastore.query.Query attribute), 39
- project (gcloud.datastore.transaction.Transaction attribute), 43
- project (gcloud.dns.zone.ManagedZone attribute), 149
- project (gcloud.pubsub.topic.Topic attribute), 88
- project (gcloud.search.index.Index attribute), 162
- PROJECT (in module gcloud.environment_vars), 17
- project_number (gcloud.storage.bucket.Bucket attribute), 68
- projection (gcloud.datastore.query.Query attribute), 39
- public_url (gcloud.storage.blob.Blob attribute), 59
- publish() (gcloud.pubsub.topic.Batch method), 87
- publish() (gcloud.pubsub.topic.Topic method), 88
- PUBSUB_EMULATOR (in module gcloud.environment_vars), 17
- pull() (gcloud.pubsub.subscription.Subscription method), 92
- put() (gcloud.datastore.batch.Batch method), 46
- put() (gcloud.datastore.client.Client method), 27
- put() (gcloud.datastore.transaction.Transaction method), 43
- put_multi() (gcloud.datastore.client.Client method), 27
- ## Q
- Query (class in gcloud.datastore.query), 37
- query() (gcloud.datastore.client.Client method), 27
- QueryJob (class in gcloud.bigquery.job), 118
- QueryPriority (class in gcloud.bigquery.job), 119
- QueryResults (class in gcloud.bigquery.query), 127
- quotas() (gcloud.dns.client.Client method), 145

quote_character (gcloud.bigquery.job.LoadTableFromStorageJob attribute), 118

R

Redirection, 14

reload() (gcloud.bigquery.dataset.Dataset method), 114

reload() (gcloud.bigquery.table.Table method), 124

reload() (gcloud.dns.changes.Changes method), 154

reload() (gcloud.dns.zone.ManagedZone method), 149

reload() (gcloud.pubsub.subscription.Subscription method), 93

reload() (gcloud.resource_manager.project.Project method), 138

reload() (gcloud.search.document.Document method), 166

reload() (gcloud.storage.acl.ACL method), 73

reload_path (gcloud.storage.acl.ACL attribute), 73

reload_path (gcloud.storage.acl.BucketACL attribute), 74

reload_path (gcloud.storage.acl.ObjectACL attribute), 74

rename_blob() (gcloud.storage.bucket.Bucket method), 69

RequestRangeNotSatisfiable, 14

reset() (gcloud.storage.acl.ACL method), 73

resource_record_set() (gcloud.dns.zone.ManagedZone method), 149

ResourceRecordSet (class in gcloud.dns.resource_record_set), 151

ResumeIncomplete, 15

rollback() (gcloud.datastore.batch.Batch method), 47

rollback() (gcloud.datastore.connection.Connection method), 29

rollback() (gcloud.datastore.transaction.Transaction method), 44

rows (gcloud.bigquery.query.QueryResults attribute), 128

run() (gcloud.bigquery.query.QueryResults method), 129

run_async_query() (gcloud.bigquery.client.Client method), 109

run_query() (gcloud.datastore.connection.Connection method), 30

run_sync_query() (gcloud.bigquery.client.Client method), 109

S

save() (gcloud.storage.acl.ACL method), 73

save_path (gcloud.storage.acl.ACL attribute), 74

save_path (gcloud.storage.acl.BucketACL attribute), 74

save_path (gcloud.storage.acl.ObjectACL attribute), 74

save_predefined() (gcloud.storage.acl.ACL method), 74

schema (gcloud.bigquery.job.LoadTableFromStorageJob attribute), 118

schema (gcloud.bigquery.query.QueryResults attribute), 129

schema (gcloud.bigquery.table.Table attribute), 124

SchemaField (class in gcloud.bigquery.table), 121

SCOPE (gcloud.bigquery.connection.Connection attribute), 110

SCOPE (gcloud.connection.Connection attribute), 9

SCOPE (gcloud.datastore.connection.Connection attribute), 28

SCOPE (gcloud.dns.connection.Connection attribute), 146

SCOPE (gcloud.pubsub.connection.Connection attribute), 84

SCOPE (gcloud.resource_manager.connection.Connection attribute), 135

SCOPE (gcloud.search.connection.Connection attribute), 160

SCOPE (gcloud.storage.connection.Connection attribute), 54

search() (gcloud.search.index.Index method), 162

self_link (gcloud.bigquery.dataset.Dataset attribute), 114

self_link (gcloud.bigquery.table.Table attribute), 125

self_link (gcloud.storage.blob.Blob attribute), 59

self_link (gcloud.storage.bucket.Bucket attribute), 69

ServerError, 15

ServiceUnavailable, 15

set_properties_from_api_repr() (gcloud.resource_manager.project.Project method), 138

size (gcloud.storage.blob.Blob attribute), 59

skip_leading_rows (gcloud.bigquery.job.LoadTableFromStorageJob attribute), 118

source_format (gcloud.bigquery.job.LoadTableFromStorageJob attribute), 118

SourceFormat (class in gcloud.bigquery.job), 119

started (gcloud.dns.changes.Changes attribute), 154

status (gcloud.dns.changes.Changes attribute), 154

status (gcloud.storage.batch.NoContent attribute), 77

storage_class (gcloud.storage.blob.Blob attribute), 60

storage_class (gcloud.storage.bucket.Bucket attribute), 69

StringValue (class in gcloud.search.document), 166

Subscription (class in gcloud.pubsub.subscription), 91

subscription() (gcloud.pubsub.topic.Topic method), 89

T

Table (class in gcloud.bigquery.table), 121

table() (gcloud.bigquery.dataset.Dataset method), 114

table_id (gcloud.bigquery.table.Table attribute), 125

table_type (gcloud.bigquery.table.Table attribute), 125

TemporaryRedirect, 15

TESTS_DATASET (in module gcloud.environment_vars), 17

TESTS_PROJECT (in module gcloud.environment_vars), 17

text_fields (gcloud.search.index.Index attribute), 163

time_created (gcloud.storage.bucket.Bucket attribute), 69

time_deleted (gcloud.storage.blob.Blob attribute), 60

- timeout_ms (gcloud.bigquery.query.QueryResults attribute), 129
 timestamp (gcloud.pubsub.message.Message attribute), 95
 TimestampValue (class in gcloud.search.document), 167
 to_protobuf() (gcloud.datastore.key.Key method), 35
 TooManyRequests, 15
 Topic (class in gcloud.pubsub.topic), 87
 topic() (gcloud.pubsub.client.Client method), 84
 total_bytes_processed (gcloud.bigquery.query.QueryResults attribute), 129
 total_rows (gcloud.bigquery.query.QueryResults attribute), 129
 Transaction (class in gcloud.datastore.transaction), 41
 transaction() (gcloud.datastore.client.Client method), 27
- ## U
- Unauthorized, 15
 undelete() (gcloud.resource_manager.project.Project method), 138
 update() (gcloud.bigquery.dataset.Dataset method), 114
 update() (gcloud.bigquery.table.Table method), 125
 update() (gcloud.resource_manager.project.Project method), 139
 updated (gcloud.storage.blob.Blob attribute), 60
 upload_from_file() (gcloud.bigquery.table.Table method), 125
 upload_from_file() (gcloud.storage.blob.Blob method), 60
 upload_from_filename() (gcloud.storage.blob.Blob method), 61
 upload_from_string() (gcloud.storage.blob.Blob method), 61
 use_query_cache (gcloud.bigquery.job.QueryJob attribute), 119
 use_query_cache (gcloud.bigquery.query.QueryResults attribute), 129
 user() (gcloud.storage.acl.ACL method), 74
 USER_AGENT (gcloud.connection.Connection attribute), 9
 UTF_8 (gcloud.bigquery.job.Encoding attribute), 116
- ## V
- value_type (gcloud.search.document.GeoValue attribute), 166
 value_type (gcloud.search.document.NumberValue attribute), 166
 value_type (gcloud.search.document.StringValue attribute), 167
 value_type (gcloud.search.document.TimestampValue attribute), 167
 versioning_enabled (gcloud.storage.bucket.Bucket attribute), 69
 view_query (gcloud.bigquery.table.Table attribute), 126
- ## W
- WRITE_APPEND (gcloud.bigquery.job.WriteDisposition attribute), 119
 write_disposition (gcloud.bigquery.job.CopyJob attribute), 115
 write_disposition (gcloud.bigquery.job.LoadTableFromStorageJob attribute), 118
 write_disposition (gcloud.bigquery.job.QueryJob attribute), 119
 WRITE_EMPTY (gcloud.bigquery.job.WriteDisposition attribute), 119
 WRITE_TRUNCATE (gcloud.bigquery.job.WriteDisposition attribute), 120
 WriteDisposition (class in gcloud.bigquery.job), 119
- ## Z
- zone() (gcloud.dns.client.Client method), 145
 zone_id (gcloud.dns.zone.ManagedZone attribute), 149