
google-cloud Documentation

Release 0.19.0

Google Cloud Platform

October 06, 2016

1	Base Client	1
2	Credentials Helpers	5
3	Base Connections	9
4	Exceptions	13
5	Environment Variables	17
6	Configuration	19
6.1	Overview	19
6.2	Authentication	19
7	Authentication	21
7.1	Overview	21
7.2	Client-Provided Authentication	21
7.3	Explicit Credentials	22
7.4	Troubleshooting	23
7.5	Advanced Customization	24
8	Long-Running Operations	27
9	Datastore Client	29
9.1	Connection	32
10	Entities	37
11	Keys	39
12	Queries	43
13	Transactions	47
14	Batches	51
15	Helpers	55
16	Storage Client	57
16.1	Connection	59

17 Blobs / Objects	61
18 Buckets	69
19 ACL	77
20 Batches	81
21 Using the API	83
21.1 Authentication / Configuration	83
21.2 Manage topics for a project	83
21.3 Publish messages to a topic	84
21.4 Manage subscriptions to topics	84
21.5 Pull messages from a subscription	86
22 Pub/Sub Client	87
22.1 Connection	88
23 Topics	91
24 Subscriptions	97
25 Message	103
26 IAM Policy	105
27 Using the API	109
27.1 Authentication / Configuration	109
27.2 Projects	109
27.3 Datasets	109
27.4 Tables	111
27.5 Jobs	112
28 BigQuery Client	119
28.1 Connection	122
29 Datasets	123
30 Jobs	127
31 Tables	133
32 Query	139
33 Schemas	143
34 Using the API	145
35 Base for Everything	147
35.1 Long-lived Defaults	147
35.2 Configuration	147
35.3 Admin API Access	147
35.4 Read-Only Mode	148
35.5 Next Step	148
36 Instance Admin API	149
36.1 List Instances	149

36.2	Instance Factory	149
36.3	Create a new Instance	149
36.4	Check on Current Operation	150
36.5	Get metadata for an existing Instance	150
36.6	Update an existing Instance	150
36.7	Delete an existing Instance	150
36.8	Next Step	150
37	Table Admin API	151
37.1	List Tables	151
37.2	Table Factory	151
37.3	Create a new Table	151
37.4	Delete an existing Table	151
37.5	List Column Families in a Table	152
37.6	Column Family Factory	152
37.7	Create a new Column Family	152
37.8	Delete an existing Column Family	152
37.9	Update an existing Column Family	152
37.10	Next Step	153
38	Data API	155
38.1	Cells vs. Columns vs. Column Families	155
38.2	Modifying Data	155
38.3	Reading Data	157
39	Client	161
40	Instance	163
41	Cluster	165
42	Table	167
43	Column Families	169
44	Bigtable Row	171
45	Bigtable Row Filters	173
46	Row Data	175
47	Resource Manager Overview	177
47.1	Authentication	178
48	Client	179
48.1	Connection	180
49	Projects	183
50	Using the API	187
50.1	Client	187
50.2	Projects	187
50.3	Project Quotas	187
50.4	Managed Zones	188
50.5	Resource Record Sets	188
50.6	Change requests	189

51 DNS Client	191
51.1 Connection	192
52 Managed Zones	193
53 Resource Record Sets	197
54 Change Sets	199
55 Using the API	201
55.1 Authentication and Configuration	201
55.2 Writing log entries	201
55.3 Retrieving log entries	202
55.4 Delete all entries for a logger	202
55.5 Manage log metrics	203
55.6 Export log entries using sinks	204
55.7 Integration with Python logging module	206
56 Python logging handler transports	209
57 Stackdriver Logging Client	211
57.1 Connection	213
58 Logger	215
59 Entries	219
60 Metrics	221
61 Sinks	223
62 Python Logging Module Handler	225
63 Python Logging Handler Sync Transport	227
64 Python Logging Handler Threaded Transport	229
65 Python Logging Handler Async Transport	231
66 Using the API	233
66.1 Authentication and Configuration	233
66.2 Reporting an exception	233
66.3 Reporting an error without an exception	234
67 Error Reporting Client	235
68 Using the API	237
68.1 Introduction	237
68.2 The Stackdriver Monitoring Client Object	237
68.3 Monitored Resource Descriptors	238
68.4 Metric Descriptors	238
68.5 Groups	239
68.6 Time Series Queries	240
69 Stackdriver Monitoring Client	241
69.1 Connection	246

70 Metric Descriptors	249
71 Monitored Resource Descriptors	251
72 Groups	253
73 Time Series Query	257
74 Time Series	263
75 Label Descriptors	265
76 Using the API	267
76.1 Authentication / Configuration	267
76.2 Methods	267
77 Translate Client	269
77.1 Connection	270
78 Using the Vision API	273
78.1 Authentication and Configuration	273
78.2 Annotating an Image	273
79 Vision Client	277
79.1 Client	277
79.2 Connection	278
80 Vision Image Properties	279
80.1 Image Properties Annotation	279
81 Vision Entity	283
81.1 Entity	283
82 Vision Feature	285
82.1 Feature	285
83 Vision Face	287
83.1 Face	287
84 Vision Image	293
84.1 Image	293
84.2 Geometry	294
84.3 Likelihood	296
85 Vision Safe Search	299
85.1 Safe Search Annotation	299
86 Using the API	301
86.1 Client	301
86.2 Methods	301
86.3 Analyze Entities	303
86.4 Analyze Sentiment	303
86.5 Annotate Text	303
87 Natural Language Client	305
87.1 Connection	306

88 Document	307
89 Natural Language Response Classes	311
89.1 Entity	311
89.2 Sentiment	312
89.3 Syntax	312
90 Getting started	315
90.1 Cloud Datastore	315
90.2 Cloud Storage	315
Python Module Index	317

Base Client

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

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

Bases: `google.cloud.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 `google.cloud.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 `google.cloud.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 `google.cloud.client.JSONClient` (*project=None, credentials=None, http=None*)
Bases: `google.cloud.client.Client`, `google.cloud.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 `google.cloud.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 *google.cloud.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.

```
google.cloud.credentials.generate_signed_url (credentials, resource, expiration,
                                             api_access_endpoint='',
                                             method='GET', content_md5=None,
                                             content_type=None, response_type=None,
                                             response_disposition=None, generation=None)
```

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

Note: Assumes `credentials` implements a `sign_blob()` method that takes bytes to sign and returns a pair of the key ID (unused here) and the signed bytes (this is abstract in the base class `oauth2client.client.AssertionCredentials`). Also assumes `credentials` has a `service_account_email` property which identifies the credentials.

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.

```
google.cloud.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',  
  'project_id': '...',  
  'private_key_id': '...',  
  'private_key': '...',  
  'client_email': '...',  
  'client_id': '...',  
  'auth_uri': '...',  
  'token_uri': '...',  
  'auth_provider_x509_cert_url': '...',  
  'client_x509_cert_url': '...' }  
}
```

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.

Base Connections

Shared implementation of connections to API servers.

```
google.cloud.connection.API_BASE_URL = 'https://www.googleapis.com'
```

The base of the API call URL.

```
class google.cloud.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.

`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.

`google.cloud.connection.DEFAULT_USER_AGENT = 'gcloud-python/0.19.0'`

The user agent for google-cloud-python requests.

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

Bases: `google.cloud.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 or list*) – A dictionary of keys and values (or list of key-value pairs) to insert into the query string of the URL.
- **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 google-cloud-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.

Return type dict or str

Returns The API response payload, either as a raw string or a dictionary if the response is valid JSON.

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* or *list*) – A dictionary of keys and values (or list of key-value pairs) 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 `google.cloud` package.

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

exception `google.cloud.exceptions.BadGateway` (*message*, *errors*=())

Bases: `google.cloud.exceptions.ServerError`

Exception mapping a '502 Bad Gateway' response.

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

Bases: `google.cloud.exceptions.ClientError`

Exception mapping a '400 Bad Request' response.

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

Bases: `google.cloud.exceptions.GoogleCloudError`

Base for 4xx responses

This class is abstract

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

Bases: `google.cloud.exceptions.ClientError`

Exception mapping a '409 Conflict' response.

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

Bases: `google.cloud.exceptions.ClientError`

Exception mapping a '403 Forbidden' response.

exception `google.cloud.exceptions.GoogleCloudError` (*message*, *errors*=())

Bases: `exceptions.Exception`

Base error class for Google Cloud 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.

`google.cloud.exceptions.GrpcRendezvous = None`

Exception class raised by gRPC stable.

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

Bases: `google.cloud.exceptions.ServerError`

Exception mapping a '500 Internal Server Error' response.

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

Bases: `google.cloud.exceptions.ClientError`

Exception mapping a '411 Length Required' response.

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

Bases: `google.cloud.exceptions.ClientError`

Exception mapping a '405 Method Not Allowed' response.

exception `google.cloud.exceptions.MethodNotImplemented` (*message*, *errors*=())

Bases: `google.cloud.exceptions.ServerError`

Exception mapping a '501 Not Implemented' response.

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

Bases: `google.cloud.exceptions.Redirection`

Exception mapping a '301 Moved Permanently' response.

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

Bases: `google.cloud.exceptions.ClientError`

Exception mapping a '404 Not Found' response.

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

Bases: `google.cloud.exceptions.Redirection`

Exception mapping a '304 Not Modified' response.

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

Bases: `google.cloud.exceptions.ClientError`

Exception mapping a '412 Precondition Failed' response.

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

Bases: `google.cloud.exceptions.GoogleCloudError`

Base for 3xx responses

This class is abstract.

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

Bases: `google.cloud.exceptions.ClientError`

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

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

Bases: `google.cloud.exceptions.Redirection`

Exception mapping a '308 Resume Incomplete' response.

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

Bases: `google.cloud.exceptions.GoogleCloudError`

Base for 5xx responses: (abstract)

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

Bases: `google.cloud.exceptions.ServerError`

Exception mapping a '503 Service Unavailable' response.

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

Bases: `google.cloud.exceptions.Redirection`

Exception mapping a '307 Temporary Redirect' response.

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

Bases: `google.cloud.exceptions.ClientError`

Exception mapping a '429 Too Many Requests' response.

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

Bases: `google.cloud.exceptions.ClientError`

Exception mapping a '401 Unauthorized' response.

`google.cloud.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 `GoogleCloudError`, or a concrete subclass.

Returns Exception specific to the error response.

Environment Variables

Comprehensive list of environment variables used in google-cloud.

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

`google.cloud.environment_vars.BIGTABLE_EMULATOR = 'BIGTABLE_EMULATOR_HOST'`
Environment variable defining host for Bigtable emulator.

`google.cloud.environment_vars.CREDENTIALS = 'GOOGLE_APPLICATION_CREDENTIALS'`
Environment variable defining location of Google credentials.

`google.cloud.environment_vars.DISABLE_GRPC = 'GOOGLE_CLOUD_DISABLE_GRPC'`
Environment variable acting as flag to disable gRPC.

To be used for APIs where both an HTTP and gRPC implementation exist.

`google.cloud.environment_vars.GCD_DATASET = 'DATASTORE_DATASET'`
Environment variable defining default dataset ID under GCD.

`google.cloud.environment_vars.GCD_HOST = 'DATASTORE_EMULATOR_HOST'`
Environment variable defining host for GCD dataset server.

`google.cloud.environment_vars.PROJECT = 'GOOGLE_CLOUD_PROJECT'`
Environment variable defining default project.

`google.cloud.environment_vars.PUBSUB_EMULATOR = 'PUBSUB_EMULATOR_HOST'`
Environment variable defining host for Pub/Sub emulator.

Configuration

6.1 Overview

Use service client objects to configure your applications.

For example:

```
>>> from google.cloud import bigquery
>>> client = bigquery.Client()
```

When creating a client in this way, the project ID will be determined by searching these locations in the following order.

- GOOGLE_CLOUD_PROJECT environment variable
- GOOGLE_APPLICATION_CREDENTIALS JSON file
- Default service configuration path from `$ gcloud beta auth application-default login`.
- Google App Engine application ID
- Google Compute Engine project ID (from metadata server)

You can override the detection of your default project by setting the `project` parameter when creating client objects.

```
>>> from google.cloud import bigquery
>>> client = bigquery.Client(project='my-project')
```

You can see what project ID a client is referencing by accessing the `project` property on the client object.

```
>>> client.project
u'my-project'
```

6.2 Authentication

The authentication credentials can be implicitly determined from the environment or directly. See [Authentication](#).

Logging in via `gcloud beta auth application-default login` will automatically configure a JSON key file with your default project ID and credentials.

Setting the `GOOGLE_APPLICATION_CREDENTIALS` and `GOOGLE_CLOUD_PROJECT` environment variables will override the automatically configured credentials.

You can change your default project ID to `my-new-default-project` by using the `gcloud` CLI tool to change the configuration.

```
$ gcloud config set project my-new-default-project
```

Authentication

7.1 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 beta auth application-default login
```

Note that this command generates credentials for client libraries. To authenticate the CLI itself, use:

```
$ gcloud auth login
```

Previously, `gcloud auth login` was used for both use cases. If your `gcloud` installation does not support the new command, please update it:

```
$ gcloud components update
```

- **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"
```

7.2 Client-Provided Authentication

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

```
from google.cloud 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](#).

7.2.1 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

7.3 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)
```

7.3.1 Google App Engine Environment

To create `credentials` just for Google App Engine:

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

7.3.2 Google Compute Engine Environment

To create `credentials` just for Google Compute Engine:

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

7.3.3 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.

7.3.4 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 `google-cloud-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()`.

7.4 Troubleshooting

7.4.1 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
 - **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**:

7.4.2 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`

7.5 Advanced Customization

Though the `google-cloud-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 `google-cloud-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.

Long-Running Operations

Wrap long-running operations returned from Google Cloud APIs.

class `google.cloud.operation.Operation` (*name*, *client*, *pb_metadata=None*, ***kw*)
 Bases: `object`

Representation of a Google API Long-Running Operation.

Parameters

- **name** (*str*) – The fully-qualified path naming the operation.
- **client** (object: must provide `_operations_stub` accessor.) – The client used to poll for the status of the operation.
- **pb_metadata** (*object*) – Instance of protobuf metadata class
- **kw** (*dict*) – caller-assigned metadata about the operation

complete

Has the operation already completed?

Return type `bool`

Returns True if already completed, else false.

classmethod `from_pb` (*op_pb*, *client*, ***kw*)

Factory: construct an instance from a protobuf.

Parameters

- **op_pb** (`google.longrunning.operations_pb2.Operation`) – Protobuf to be parsed.
- **client** (object: must provide `_operations_stub` accessor.) – The client used to poll for the status of the operation.
- **kw** (*dict*) – caller-assigned metadata about the operation

Return type `Operation`

Returns new instance, with attributes based on the protobuf.

poll ()

Check if the operation has finished.

Return type `bool`

Returns A boolean indicating if the current operation has completed.

Raises `ValueError` if the operation has already completed.

target = None

Instance associated with the operations: callers may set.

Datastore Client

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

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

Bases: `google.cloud.client.Client`, `google.cloud.client._ClientProjectMixin`

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** (`google.cloud.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 `google.cloud.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 `google.cloud.datastore.batch.Batch`.

current_batch

Currently-active batch.

Return type `google.cloud.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 `google.cloud.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** (`google.cloud.datastore.key.Key`) – The key to be deleted from the datastore.

delete_multi (*keys*)

Delete keys from the Cloud Datastore.

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

get (*key, missing=None, deferred=None, transaction=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** (`google.cloud.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.
- **transaction** (`Transaction`) – (Optional) Transaction to use for read consistency. If not passed, uses current transaction, if set.

Return type `google.cloud.datastore.entity.Entity` or `NoneType`

Returns The requested entity if it exists.

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

Retrieve entities, along with their attributes.

Parameters

- **keys** (list of `google.cloud.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.
- **transaction** (*Transaction*) – (Optional) Transaction to use for read consistency. If not passed, uses current transaction, if set.

Return type list of `google.cloud.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 `google.cloud.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` (`google.cloud.datastore.entity.Entity`) – The entity to be saved to the datastore.

put_multi (*entities*)

Save entities in the Cloud Datastore.

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

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

query (***kwargs*)

Proxy to `google.cloud.datastore.query.Query`.

Passes our project.

Using query to search a datastore:

```
>>> from google.cloud import datastore
>>> client = datastore.Client()
>>> query = client.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 `kwargs` (*dict*) – Parameters for initializing and instance of `google.cloud.datastore.query.Query`.

Return type `google.cloud.datastore.query.Query`

Returns An instance of `google.cloud.datastore.query.Query`

transaction()

Proxy to `google.cloud.datastore.transaction.Transaction`.

9.1 Connection

Connections to Google Cloud Datastore API servers.

class `google.cloud.datastore.connection.Connection` (*credentials=None, http=None*)

Bases: `google.cloud.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 it 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 = `'https://datastore.googleapis.com'`

The base of the API call URL.

API_URL_TEMPLATE = `'{api_base}/{api_version}/projects/{project}:{method}'`

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/datastore',)`

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 `google.cloud.datastore._generated.entity_pb2.Key`) – The keys for which the backend should allocate IDs.

Return type list of `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.

Return type str

Returns The API URL created.

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 (`google.cloud.datastore._generated.entity_pb2.Key` as input and `google.cloud.datastore._generated.entity_pb2.Entity` as output). It is used under the hood in `Client.get()`:

```
>>> from google.cloud 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 `google.cloud.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 `google.cloud.datastore._generated.entity_pb2.Entity` and `deferred` is a list of `google.cloud.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 `google.cloud.datastore.query.Query.fetch()` method.

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

Parameters

- **project** (*string*) – The project over which to run the query.
- **query_pb** (`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`.

Return type `tuple`

Returns Four-tuple containing the entities returned, the end cursor of the query, a `more_results` enum and a count of the number of skipped results.

`google.cloud.datastore.connection.DATASTORE_API_HOST = 'datastore.googleapis.com'`
Datastore API request host.

Entities

Class for representing a single entity in the Cloud Datastore.

```
class google.cloud.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 `google.cloud.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 `google.cloud.datastore.get()` to retrieve an existing entity.

```
>>> from google.cloud import datastore
>>> client = datastore.Client()
>>> client.get(key)
<Entity[{'kind': 'EntityKind', id: 1234}] {'property': 'value'}>
```

You can 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** (*google.cloud.datastore.key.Key*) – Optional key to be set on entity.
- **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

Returns The set of fields excluded from indexes.

kind

Get the kind of the current entity.

Note: This relies entirely on the *google.cloud.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 Google Cloud Datastore keys.

```
class google.cloud.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** (*google.cloud.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 `google.cloud.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 `google.cloud.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 `google.cloud.datastore._generated.entity_pb2.Key`

Returns The protobuf representing the key.

Queries

Create / interact with Google Cloud Datastore queries.

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

Bases: `object`

Represent the state of a given execution of a Query.

Parameters

- **query** (*google.cloud.datastore.query.Query*) – Query object holding permanent configuration (i.e. things that don't change on with each page in a results set).
- **client** (*google.cloud.datastore.client.Client*) – The client used to make a request.
- **limit** (*integer*) – (Optional) Limit the number of results returned.
- **offset** (*integer*) – (Optional) 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)

Returns The next page of results.

```
class google.cloud.datastore.query.Query (client, kind=None, project=None, namespace=None,
ancestor=None, filters=(), projection=(), order=(),
distinct_on=())
```

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** (*google.cloud.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** (*google.cloud.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.
- **distinct_on** (*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 google.cloud import datastore
>>> client = datastore.Client()
>>> query = client.query(kind='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, datetime.datetime, google.cloud.datastore.key.Key*) – 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`

Returns The ancestor for the query.

distinct_on

Names of fields used to group query results.

Return type `sequence of string`

Returns The “distinct on” fields set on the query.

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 google.cloud import datastore
>>> client = datastore.Client()
>>> query = client.query(kind='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** (*google.cloud.datastore.client.Client*) – client used to connect to datastore. If not supplied, uses the query’s value.

Return type *Iterator*

Returns The iterator for the query.

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.

Returns The filters set on the query.

key_filter (*key, operator='='*)

Filter on a key.

Parameters

- **key** (*google.cloud.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*

Returns The kind for the query.

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

Returns The order(s) set on the query.

project

Get the project for this Query.

Return type `str`

Returns The project for the query.

projection

Fields names returned by the query.

Return type sequence of string

Returns Names of fields in query results.

Transactions

Create / interact with Google Cloud Datastore transactions.

```
class google.cloud.datastore.transaction.Transaction(client)
    Bases: google.cloud.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 or upsert) into the same mutation, and execute those within a transaction:

```
>>> from google.cloud import datastore
>>> client = datastore.Client()
>>> with client.transaction():
...     client.put_multi([entity1, entity2])
```

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

```
>>> with client.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 client.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 client.transaction():
...     entity = datastore.Entity(key=client.key('Thing'))
...     client.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 client.transaction():
...     entity = datastore.Entity(key=client.key('Thing'))
...     client.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 = client.transaction()
>>> transaction.begin()
>>>
>>> entity = datastore.Entity(key=client.key('Thing'))
>>> transaction.put(entity)
>>>
>>> if error:
...     transaction.rollback()
... else:
...     transaction.commit()
```

Parameters `client` (*google.cloud.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 *google.cloud.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 `google.cloud.datastore.transaction.Transaction` or None

Returns The current transaction (if any are active).

delete (*key*)

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

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

Raises `ValueError` if the batch is not in progress, 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 iterable

Returns The list of `_generated.datastore_pb2.Mutation` protobufs 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` 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` (`google.cloud.datastore.entity.Entity`) – the entity to be saved.

Raises `ValueError` if the batch is not in progress, 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 `google.cloud.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 google.cloud import datastore
>>> client = datastore.Client()
>>> batch = client.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:
...     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:
...     do_some_work(batch)
...     raise Exception() # rolls back
```

Parameters `client` (*google.cloud.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 `google.cloud.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.

Raises `ValueError` if the batch is not in progress.

`connection`

Getter for connection over which the batch will run.

Return type `google.cloud.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` (`google.cloud.datastore.key.Key`) – the key to be deleted.

Raises `ValueError` if the batch is not in progress, 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 `iterable`

Returns The list of `_generated.datastore_pb2.Mutation` protobufs 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` 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` (`google.cloud.datastore.entity.Entity`) – the entity to be saved.

Raises `ValueError` if the batch is not in progress, 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 `google.cloud.datastore.transaction.Transaction`.

Raises `ValueError` if the batch is not in progress.

Helpers

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

The non-private functions are part of the API.

`google.cloud.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` (`google.cloud.datastore._generated.entity_pb2.Entity`) – The Protobuf representing the entity.

Return type `google.cloud.datastore.entity.Entity`

Returns The entity derived from the protobuf.

`google.cloud.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` (`google.cloud.datastore._generated.entity_pb2.Key`) – The Protobuf representing the key.

Return type `google.cloud.datastore.key.Key`

Returns a new `Key` instance

Storage Client

Client for interacting with the Google Cloud Storage API.

class `google.cloud.storage.client.Client` (*project=None, credentials=None, http=None*)
Bases: `google.cloud.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 `google.cloud.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 `google.cloud.storage.bucket.Bucket`

Returns The bucket object created.

connection

Get connection or batch on the client.

Return type `google.cloud.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 `google.cloud.exceptions.Conflict`.

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

Return type `google.cloud.storage.bucket.Bucket`

Returns The newly created bucket.

current_batch

Currently-active batch.

Return type `google.cloud.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 `google.cloud.storage.exceptions.NotFound`.

For example:

```
>>> try:
>>>     bucket = client.get_bucket('my-bucket')
>>> except google.cloud.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 `google.cloud.storage.bucket.Bucket`

Returns The bucket matching the name provided.

Raises `google.cloud.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 `google.cloud.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('doesnt-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 `google.cloud.storage.bucket.Bucket`

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

16.1 Connection

Create / interact with Google Cloud Storage connections.

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

Bases: `google.cloud.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 `google.cloud.storage.blob.Blob` (*name, bucket, chunk_size=None*)

Bases: `google.cloud.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** (*google.cloud.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** (*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 `google.cloud.exceptions.NotFound` (propagated from `google.cloud.storage.bucket.Bucket.delete_blob()`).

download_as_string (*encryption_key=None, client=None*)

Download the contents of this blob as a string.

Parameters

- **encryption_key** (*str* or *bytes*) – Optional 32 byte encryption key for customer-supplied encryption.
- **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 `google.cloud.exceptions.NotFound`

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

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

Note:

If the server-set property, `media_link`, is not yet initialized, makes an additional API request to load it.

Downloading a file that has been encrypted with a **customer-supplied** encryption key:

```
>>> from google.cloud import storage
>>> from google.cloud.storage import Blob

>>> client = storage.Client(project='my-project')
>>> bucket = client.get_bucket('my-bucket')
>>> encryption_key = 'aa426195405adee2c8081bb9e7e74b19'
>>> blob = Blob('secure-data', bucket)
>>> with open('/tmp/my-secure-file', 'wb') as file_obj:
>>>     blob.download_to_file(file_obj,
...                           encryption_key=encryption_key)
```

The `encryption_key` should be a str or bytes with a length of at least 32.

Parameters

- **file_obj** (*file*) – A file handle to which to write the blob’s data.
- **encryption_key** (*str or bytes*) – Optional 32 byte encryption key for customer-supplied encryption.
- **client** (*Client* or *NoneType*) – Optional. The client to use. If not passed, falls back to the `client` stored on the blob’s bucket.

Raises `google.cloud.exceptions.NotFound`

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

Download the contents of this blob into a named file.

Parameters

- **filename** (*string*) – A filename to be passed to `open`.
- **encryption_key** (*str or bytes*) – Optional 32 byte encryption key for customer-supplied encryption.
- **client** (*Client* or *NoneType*) – Optional. The client to use. If not passed, falls back to the `client` stored on the blob’s bucket.

Raises `google.cloud.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** (*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** (*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** (*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*, *encryption_key=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.

Uploading a file with a customer-supplied encryption key:

```
>>> from google.cloud import storage
>>> from google.cloud.storage import Blob

>>> client = storage.Client(project='my-project')
>>> bucket = client.get_bucket('my-bucket')
>>> encryption_key = 'aa426195405adee2c8081bb9e7e74b19'
>>> blob = Blob('secure-data', bucket)
>>> with open('my-file', 'rb') as my_file:
>>>     blob.upload_from_file(my_file,
...                           encryption_key=encryption_key)
```

The `encryption_key` should be a str or bytes with a length of at least 32.

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.)
- **encryption_key** (*str or bytes*) – Optional 32 byte encryption key for customer-supplied encryption.
- **content_type** (string or `NoneType`) – Optional type of content being uploaded.
- **num_retries** (*integer*) – Number of upload retries. Defaults to 6.
- **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; `google.cloud.exceptions.GoogleCloudError` if the upload response returns an error status.

upload_from_filename (*filename*, *content_type=None*, *encryption_key=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.
- **encryption_key** (*str* or *bytes*) – Optional 32 byte encryption key for customer-supplied encryption.
- **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'*, *encryption_key=None*, *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'`.
- **encryption_key** (*str* or *bytes*) – Optional 32 byte encryption key for customer-supplied encryption.
- **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 Google Cloud Storage buckets.

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

A class representing a Bucket on Cloud Storage.

Parameters

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

Return type *google.cloud.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 *google.cloud.exceptions.Conflict*.

This implements “storage.buckets.insert”.

Parameters **client** (*Client* or *NoneType*) – Optional. The client to use. If not passed, falls back to the `client` stored on the current 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 `google.cloud.exceptions.NotFound`. If the bucket is not empty (and `force=False`), will raise `google.cloud.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** (*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 `google.cloud.exceptions.NotFound`.

For example:

```
>>> from google.cloud.exceptions import NotFound
>>> from google.cloud 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** (*Client* or *NoneType*) – Optional. The client to use. If not passed, falls back to the `client` stored on the current bucket.

Raises `google.cloud.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 `google.cloud.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 `google.cloud.exceptions.NotFound`; otherwise, the exception is propagated.
- **client** (*Client* or `NoneType`) – Optional. The client to use. If not passed, falls back to the `client` stored on the current bucket.

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

Return type `google.cloud.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** (*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** (*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** (*google.cloud.storage.blob.Blob*) – The blob to be renamed.
- **new_name** (*string*) – The new name for this blob.
- **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.

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

```
>>> from google.cloud 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 `google.cloud.storage.acl.ACL.save()` method:

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

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

Bases: `object`

Container class representing a list of access controls.

PREDEFINED_JSON_ACLS = `frozenset(['publicRead', 'bucketOwnerFullControl', 'bucketOwnerRead', 'projectPrivate'])`

See: <https://cloud.google.com/storage/docs/access-control#predefined-acl>

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` (`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.

reload (*client=None*)

Reload the ACL data from Cloud Storage.

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

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

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_JSON_ACLS* or *PREDEFINED_XML_ACLS* (which will be aliased to the corresponding JSON name). If passed, *acl* must be *None*.
- **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 *google.cloud.storage.acl*.**BucketACL** (*bucket*)

Bases: *google.cloud.storage.acl.ACL*

An ACL specifically for a bucket.

Parameters **bucket** (*google.cloud.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 *google.cloud.storage.acl*.**DefaultObjectACL** (*bucket*)

Bases: *google.cloud.storage.acl.BucketACL*

A class representing the default object ACL for a bucket.

class *google.cloud.storage.acl*.**ObjectACL** (*blob*)

Bases: *google.cloud.storage.acl.ACL*

An ACL specifically for a Cloud Storage object / blob.

Parameters **blob** (*google.cloud.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 `google.cloud.storage.batch.Batch` (*client*)

Bases: `google.cloud.storage.connection.Connection`

Proxy an underlying connection, batching up change operations.

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

current ()

Return the topmost batch, or None.

finish ()

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

Return type list of tuples

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

class `google.cloud.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** (*str*) – HTTP method
- **uri** (*str*) – URI for HTTP request
- **headers** (*dict*) – HTTP headers
- **body** (*str or None*) – HTTP payload

class `google.cloud.storage.batch.NoContent`

Bases: `object`

Emulate an HTTP ‘204 No Content’ response.

Using the API

21.1 Authentication / Configuration

- Use *Client* objects to configure your applications.
- In addition to any authentication configuration, you should also set the `GOOGLE_CLOUD_PROJECT` environment variable for the project you'd like to interact with. If you are Google App Engine or Google Compute Engine this will be detected automatically.
- The library now enables the `gRPC` transport for the `pubsub` API by default, assuming that the required dependencies are installed and importable. To *disable* this transport, set the `GOOGLE_CLOUD_DISABLE_GRPC` environment variable to a non-empty string, e.g.: `$ export GOOGLE_CLOUD_DISABLE_GRPC=true`.
- *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 `GOOGLE_CLOUD_PROJECT` environment variables, create a *Client*

```
>>> from google.cloud import pubsub
>>> client = pubsub.Client()
```

21.2 Manage topics for a project

List topics for the default project:

```
topics, token = client.list_topics() # API request
while True:
    for topic in topics:
        do_something_with(topic)
    if token is None:
        break
    topics, token = client.list_topics(page_token=token) # API request
```

Create a new topic for the default project:

```
topic = client.topic(TOPIC_NAME)
topic.create() # API request
```

Check for the existence of a topic:

```
assert not topic.exists() # API request
topic.create() # API request
assert topic.exists() # API request
```

Delete a topic:

```
assert topic.exists() # API request
topic.delete()
assert not topic.exists() # API request
```

Fetch the IAM policy for a topic:

```
policy = topic.get_iam_policy() # API request
```

Update the IAM policy for a topic:

```
ALL_USERS = policy.all_users()
policy.viewers.add(ALL_USERS)
LOGS_GROUP = policy.group('cloud-logs@google.com')
policy.editors.add(LOGS_GROUP)
new_policy = topic.set_iam_policy(policy) # API request
```

Test permissions allowed by the current IAM policy on a topic:

```
from google.cloud.pubsub.iam import OWNER_ROLE, EDITOR_ROLE, VIEWER_ROLE
TO_CHECK = [OWNER_ROLE, EDITOR_ROLE, VIEWER_ROLE]
ALLOWED = topic.check_iam_permissions(TO_CHECK)
assert set(ALLOWED) == set(TO_CHECK)
```

21.3 Publish messages to a topic

Publish a single message to a topic, without attributes:

```
topic.publish(b'This is the message payload') # API request
```

Publish a single message to a topic, with attributes:

```
topic.publish(b'Another message payload', extra='EXTRA') # API request
```

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

```
with topic.batch() as batch:
    batch.publish(PAYLOAD1)
    batch.publish(PAYLOAD2, extra=EXTRA)
```

Note: The only API request happens during the `__exit__()` of the topic used as a context manager, and only if the block exits without raising an exception.

21.4 Manage subscriptions to topics

List all subscriptions for the default project:

```

subscriptions, token = client.list_subscriptions() # API request
while True:
    for subscription in subscriptions:
        do_something_with(subscription)
    if token is None:
        break
    subscriptions, token = client.list_subscriptions(
        page_token=token) # API request

```

List subscriptions for a topic:

```

subscriptions, token = topic.list_subscriptions() # API request
while True:
    for subscription in subscriptions:
        do_something_with(subscription)
    if token is None:
        break
    subscriptions, token = topic.list_subscriptions(
        page_token=token) # API request

```

Create a new pull subscription for a topic, with defaults:

```
sub_defaults = topic.subscription(SUB_DEFAULTS)
```

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

```
sub_ack90 = topic.subscription(SUB_ACK90, ack_deadline=90)
```

Create a new push subscription for a topic:

```
subscription = topic.subscription(SUB_PUSH, push_endpoint=PUSH_URL)
subscription.create() # API request
```

Check for the existence of a subscription:

```
assert subscription.exists() # API request
```

Convert a pull subscription to push:

```
subscription.modify_push_configuration(
    push_endpoint=PUSH_URL) # API request
```

Convert a push subscription to pull:

```
subscription.modify_push_configuration(push_endpoint=None) # API request
```

Re-synchronize a subscription with the back-end:

```
subscription.reload() # API request
```

Fetch the IAM policy for a subscription

```
policy = subscription.get_iam_policy() # API request
```

Update the IAM policy for a subscription:

```

ALL_USERS = policy.all_users()
policy.viewers.add(ALL_USERS)
LOGS_GROUP = policy.group('cloud-logs@google.com')
policy.editors.add(LOGS_GROUP)
new_policy = subscription.set_iam_policy(policy) # API request

```

Test permissions allowed by the current IAM policy on a subscription:

```
from google.cloud.pubsub.iam import OWNER_ROLE, EDITOR_ROLE, VIEWER_ROLE
TO_CHECK = [OWNER_ROLE, EDITOR_ROLE, VIEWER_ROLE]
ALLOWED = subscription.check_iam_permissions(TO_CHECK)
assert set(ALLOWED) == set(TO_CHECK)
```

Delete a subscription:

```
subscription.delete() # API request
```

21.5 Pull messages from a subscription

Fetch pending messages for a pull subscription:

```
pulled = subscription.pull(max_messages=2)
```

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

```
for ack_id, message in pulled:
    try:
        do_something_with(message)
    except ApplicationException as e:
        log_exception(e)
    else:
        subscription.acknowledge([ack_id])
```

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

```
pulled = subscription.pull(return_immediately=True)
```

Update the acknowledgement deadline for pulled messages:

```
for ack_id, _ in pulled:
    subscription.modify_ack_deadline(ack_id, 90) # API request
```

Fetch pending messages, acknowledging those whose processing doesn't raise an error:

```
from google.cloud.pubsub.subscription import AutoAck
with AutoAck(subscription, max_messages=10) as ack:
    for ack_id, message in list(ack.items()):
        try:
            do_something_with(message)
        except Exception: # pylint: disable=broad-except
            del ack[ack_id]
```

Note: The pull API request occurs at entry to the with block, and the acknowledge API request occurs at the end, passing only the ack_ids which haven't been deleted from ack

Pub/Sub Client

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

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

`iam_policy_api`

Helper for IAM policy-related API calls.

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

List subscriptions for the project associated with this client.

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

Example:

```
subscriptions, token = client.list_subscriptions() # API request
while True:
    for subscription in subscriptions:
        do_something_with(subscription)
    if token is None:
        break
subscriptions, token = client.list_subscriptions(
    page_token=token) # API request
```

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 *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>

Example:

```
topics, token = client.list_topics() # API request
while True:
    for topic in topics:
        do_something_with(topic)
    if token is None:
        break
topics, token = client.list_topics(page_token=token) # API request
```

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 *google.cloud.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`).

publisher_api

Helper for publisher-related API calls.

subscriber_api

Helper for subscriber-related API calls.

topic (*name, timestamp_messages=False*)

Creates a topic bound to the current client.

Example:

Parameters

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

Return type *google.cloud.pubsub.topic.Topic*

Returns *Topic* created with the current client.

22.1 Connection

Create / interact with Google Cloud Pub/Sub connections.

class `google.cloud.pubsub.connection.Connection` (*credentials=None, http=None*)

Bases: `google.cloud.connection.JSONConnection`

A connection to Google Cloud Pub/Sub 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://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 or list*) – A dictionary of keys and values (or list of key-value pairs) 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.

`google.cloud.pubsub.connection.PUBSUB_API_HOST` = `'pubsub.googleapis.com'`

Pub / Sub API request host.

Topics

Define API Topics.

class `google.cloud.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** (*google.cloud.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.

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

Creates a subscription bound to the current topic.

Example: pull-mode subscription, default parameter values

```
sub_defaults = topic.subscription(SUB_DEFAULTS)
```

Example: pull-mode subscription, override `ack_deadline` default

```
sub_ack90 = topic.subscription(SUB_ACK90, ack_deadline=90)
```

Example: push-mode subscription

```
subscription = topic.subscription(SUB_PUSH, push_endpoint=PUSH_URL)
subscription.create() # API request
```

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.

Return type `Subscription`

Returns The subscription created with the passed in arguments.

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** (*google.cloud.pubsub.client.Client*) – Client which holds credentials and project configuration for the topic.

Return type *google.cloud.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.

project

Project bound to the topic.

full_name

Fully-qualified name used in topic / subscription APIs

create (*client=None*)

API call: create the topic via a PUT request

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

Example:

```
topic = client.topic(TOPIC_NAME)
topic.create()           # API request
```

Parameters `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>

Example:

```
assert not topic.exists() # API request
topic.create()           # API request
assert topic.exists()    # API request
```

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

Return type `bool`

Returns Boolean indicating existence of the 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>

Example:

```

assert topic.exists()      # API request
topic.delete()
assert not topic.exists()  # API request

```

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

Example without message attributes:

```

topic.publish(b'This is the message payload')      # API request

```

With message attributes:

```

topic.publish(b'Another message payload', extra='EXTRA')  # API request

```

Parameters

- **message** (*bytes*) – the message payload
- **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

batch (*client=None*)

Return a batch to use as a context manager.

Example:

```

with topic.batch() as batch:
    batch.publish(PAYLOAD1)
    batch.publish(PAYLOAD2, extra=EXTRA)

```

Note: The only API request happens during the `__exit__()` of the topic used as a context manager, and only if the block exits without raising an exception.

Parameters **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.

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

List subscriptions for the project associated with this client.

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

Example:

```

subscriptions, token = topic.list_subscriptions() # API request
while True:
    for subscription in subscriptions:
        do_something_with(subscription)
    if token is None:
        break
subscriptions, token = topic.list_subscriptions(
    page_token=token) # API request

```

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

Return type tuple, (list, str)

Returns list of *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`).

get_iam_policy (*client=None*)

Fetch the IAM policy for the topic.

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

Example:

```
policy = topic.get_iam_policy() # API request
```

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

Return type *google.cloud.pubsub.iam.Policy*

Returns policy created from the resource returned by the `getIamPolicy` API request.

set_iam_policy (*policy, client=None*)

Update the IAM policy for the topic.

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

Example:

```

ALL_USERS = policy.all_users()
policy.viewers.add(ALL_USERS)
LOGS_GROUP = policy.group('cloud-logs@google.com')
policy.editors.add(LOGS_GROUP)
new_policy = topic.set_iam_policy(policy) # API request

```

Parameters

- **policy** (*google.cloud.pubsub.iam.Policy*) – the new policy, typically fetched via `get_iam_policy()` and updated in place.
- **client** (*Client* or *NoneType*) – the client to use. If not passed, falls back to the `client` stored on the current batch.

Return type `google.cloud.pubsub.iam.Policy`

Returns updated policy created from the resource returned by the `setIamPolicy` API request.

check_iam_permissions (*permissions*, *client=None*)

Verify permissions allowed for the current user.

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

Example:

```
from google.cloud.pubsub.iam import OWNER_ROLE, EDITOR_ROLE, VIEWER_ROLE
TO_CHECK = [OWNER_ROLE, EDITOR_ROLE, VIEWER_ROLE]
ALLOWED = topic.check_iam_permissions(TO_CHECK)
assert set(ALLOWED) == set(TO_CHECK)
```

Parameters

- **permissions** (*list of string*) – list of permissions to be tested
- **client** (*Client* or *NoneType*) – the client to use. If not passed, falls back to the `client` stored on the current batch.

Return type sequence of string

Returns subset of permissions allowed by current IAM policy.

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

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

commit (*client=None*)

Send saved messages as a single API call.

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

Subscriptions

Define API Subscriptions.

```
class google.cloud.pubsub.subscription.Subscription (name, topic=None,
                                                    ack_deadline=None,
                                                    push_endpoint=None,
                                                    client=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** (*google.cloud.pubsub.topic.Topic* or `NoneType`) – the topic to which the subscription belongs; if `None`, the subscription's topic has been deleted.
- **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.
- **client** (*Client* or `NoneType`) – the client to use. If not passed, falls back to the `client` stored on the 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** (*google.cloud.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 *google.cloud.pubsub.subscription.Subscription*

Returns Subscription parsed from *resource*.

project

Project bound to the subscription.

full_name

Fully-qualified name used in subscription APIs

path

URL path for the subscription's APIs

auto_ack (*return_immediately=False, max_messages=1, client=None*)*AutoAck* factory**Parameters**

- **return_immediately** (*boolean*) – passed through to *Subscription.pull()*
- **max_messages** (*int*) – passed through to *Subscription.pull()*
- **client** (*Client* or *NoneType*) – passed through to *Subscription.pull()* and *Subscription.acknowledge()*.

Return type *AutoAck***Returns** the instance created for the given `ack_id` and message**create** (*client=None*)

API call: create the subscription via a PUT request

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

Example:

```
subscription = topic.subscription(SUB_NAME)
subscription.create() # API request
```

Parameters **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>

Example:

```
assert subscription.exists() # API request
```

Parameters **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 `bool`**Returns** Boolean indicating existence of the subscription.**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>

Example:

```
subscription.reload() # API request
```

Parameters **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>

Example:

```
subscription.delete() # API request
```

Parameters **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>

Example:

```
subscription.modify_push_configuration(push_endpoint=None) # API request
```

```
subscription.modify_push_configuration(
    push_endpoint=PUSH_URL) # API request
```

Parameters

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

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>

Example:

```
pulled = subscription.pull(max_messages=2)
```

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** (*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 `Message`.

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>

Example:

```

for ack_id, message in pulled:
    try:
        do_something_with(message)
    except ApplicationException as e:
        log_exception(e)
    else:
        subscription.acknowledge([ack_id])

```

Parameters

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

modify_ack_deadline (*ack_ids, ack_deadline, client=None*)

API call: update acknowledgement deadline for a retrieved message.

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

Parameters

- **ack_ids** (*list of string*) – ack IDs of messages being updated
- **ack_deadline** (*int*) – new deadline for the message, in seconds
- **client** (*Client* or *NoneType*) – the client to use. If not passed, falls back to the `client` stored on the current subscription’s topic.

get_iam_policy (*client=None*)

Fetch the IAM policy for the subscription.

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

Example:

```
policy = subscription.get_iam_policy() # API request
```

Parameters **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 *google.cloud.pubsub.iam.Policy*

Returns policy created from the resource returned by the `getIamPolicy` API request.

set_iam_policy (*policy, client=None*)

Update the IAM policy for the subscription.

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

Example:

```

ALL_USERS = policy.all_users()
policy.viewers.add(ALL_USERS)
LOGS_GROUP = policy.group('cloud-logs@google.com')
policy.editors.add(LOGS_GROUP)
new_policy = subscription.set_iam_policy(policy) # API request

```

Parameters

- **policy** (*google.cloud.pubsub.iam.Policy*) – the new policy, typically fetched via `get_iam_policy()` and updated in place.

- **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 `google.cloud.pubsub.iam.Policy`

Returns updated policy created from the resource returned by the `setIamPolicy` API request.

check_iam_permissions (*permissions*, *client=None*)

Verify permissions allowed for the current user.

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

Example:

```
from google.cloud.pubsub.iam import OWNER_ROLE, EDITOR_ROLE, VIEWER_ROLE
TO_CHECK = [OWNER_ROLE, EDITOR_ROLE, VIEWER_ROLE]
ALLOWED = subscription.check_iam_permissions(TO_CHECK)
assert set(ALLOWED) == set(TO_CHECK)
```

Parameters

- **permissions** (*list of string*) – list of permissions to be tested
- **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 sequence of string

Returns subset of permissions allowed by current IAM policy.

class `google.cloud.pubsub.subscription.AutoAck` (*subscription*, *return_immediately=False*, *max_messages=1*, *client=None*)

Bases: `dict`

Wrapper for `Subscription.pull()` results.

Mapping, tracks messages still-to-be-acknowledged.

When used as a context manager, acknowledges all messages still in the mapping on `__exit__`. When processing the pulled messages, application code MUST delete messages from the `AutoAck` mapping which are not successfully processed, e.g.:

Parameters

- **subscription** (*Subscription*) – subscription to be pulled.
- **return_immediately** (*boolean*) – passed through to `Subscription.pull()`
- **max_messages** (*int*) – passed through to `Subscription.pull()`
- **client** (*Client* or *NoneType*) – passed through to `Subscription.pull()` and `Subscription.acknowledge()`.

Message

Define API Topics.

class `google.cloud.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

Return type `Message`

Returns The message created from the response.

service_timestamp

Return server-set timestamp.

Return type `string`

Returns timestamp (in UTC timezone) in RFC 3339 format

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.

IAM Policy

PubSub API IAM policy definitions

For allowed roles / permissions, see: https://cloud.google.com/pubsub/access_control#permissions

`google.cloud.pubsub.iam.OWNER_ROLE = 'roles/owner'`

Generic role implying all rights to an object.

`google.cloud.pubsub.iam.EDITOR_ROLE = 'roles/editor'`

Generic role implying rights to modify an object.

`google.cloud.pubsub.iam.VIEWER_ROLE = 'roles/viewer'`

Generic role implying rights to access an object.

`google.cloud.pubsub.iam.PUBSUB_ADMIN_ROLE = 'roles/pubsub.admin'`

Role implying all rights to an object.

`google.cloud.pubsub.iam.PUBSUB_EDITOR_ROLE = 'roles/pubsub.editor'`

Role implying rights to modify an object.

`google.cloud.pubsub.iam.PUBSUB_VIEWER_ROLE = 'roles/pubsub.viewer'`

Role implying rights to access an object.

`google.cloud.pubsub.iam.PUBSUB_PUBLISHER_ROLE = 'roles/pubsub.publisher'`

Role implying rights to publish to a topic.

`google.cloud.pubsub.iam.PUBSUB_SUBSCRIBER_ROLE = 'roles/pubsub.subscriber'`

Role implying rights to subscribe to a topic.

`google.cloud.pubsub.iam.PUBSUB_TOPICS_CONSUME = 'pubsub.topics.consume'`

Permission: consume events from a subscription.

`google.cloud.pubsub.iam.PUBSUB_TOPICS_CREATE = 'pubsub.topics.create'`

Permission: create topics.

`google.cloud.pubsub.iam.PUBSUB_TOPICS_DELETE = 'pubsub.topics.delete'`

Permission: delete topics.

`google.cloud.pubsub.iam.PUBSUB_TOPICS_GET = 'pubsub.topics.get'`

Permission: retrieve topics.

`google.cloud.pubsub.iam.PUBSUB_TOPICS_GET_IAM_POLICY = 'pubsub.topics.getIamPolicy'`

Permission: retrieve subscription IAM policies.

`google.cloud.pubsub.iam.PUBSUB_TOPICS_LIST = 'pubsub.topics.list'`

Permission: list topics.

`google.cloud.pubsub.iam.PUBSUB_TOPICS_SET_IAM_POLICY = 'pubsub.topics.setIamPolicy'`
 Permission: update subscription IAM policies.

`google.cloud.pubsub.iam.PUBSUB_SUBSCRIPTIONS_CONSUME = 'pubsub.subscriptions.consume'`
 Permission: consume events from a subscription.

`google.cloud.pubsub.iam.PUBSUB_SUBSCRIPTIONS_CREATE = 'pubsub.subscriptions.create'`
 Permission: create subscriptions.

`google.cloud.pubsub.iam.PUBSUB_SUBSCRIPTIONS_DELETE = 'pubsub.subscriptions.delete'`
 Permission: delete subscriptions.

`google.cloud.pubsub.iam.PUBSUB_SUBSCRIPTIONS_GET = 'pubsub.subscriptions.get'`
 Permission: retrieve subscriptions.

`google.cloud.pubsub.iam.PUBSUB_SUBSCRIPTIONS_GET_IAM_POLICY = 'pubsub.subscriptions.getIamPolicy'`
 Permission: retrieve subscription IAM policies.

`google.cloud.pubsub.iam.PUBSUB_SUBSCRIPTIONS_LIST = 'pubsub.subscriptions.list'`
 Permission: list subscriptions.

`google.cloud.pubsub.iam.PUBSUB_SUBSCRIPTIONS_SET_IAM_POLICY = 'pubsub.subscriptions.setIamPolicy'`
 Permission: update subscription IAM policies.

`google.cloud.pubsub.iam.PUBSUB_SUBSCRIPTIONS_UPDATE = 'pubsub.subscriptions.update'`
 Permission: update subscriptions.

class `google.cloud.pubsub.iam.Policy` (*etag=None, version=None*)
 Bases: `object`

Combined IAM Policy / Bindings.

See: <https://cloud.google.com/pubsub/reference/rest/Shared.Types/Policy> <https://cloud.google.com/pubsub/reference/rest/Shared.Types/Policy>

Parameters

- **etag** (*string*) – ETag used to identify a unique of the policy
- **version** (*int*) – unique version of the policy

static user (*email*)

Factory method for a user member.

Parameters **email** (*string*) – E-mail for this particular user.

Return type `string`

Returns A member string corresponding to the given user.

static service_account (*email*)

Factory method for a service account member.

Parameters **email** (*string*) – E-mail for this particular service account.

Return type `string`

Returns A member string corresponding to the given service account.

static group (*email*)

Factory method for a group member.

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

Return type `string`

Returns A member string corresponding to the given group.

static domain (*domain*)

Factory method for a domain member.

Parameters **domain** (*string*) – The domain for this member.

Return type *string*

Returns A member string corresponding to the given domain.

static all_users ()

Factory method for a member representing all users.

Return type *string*

Returns A member string representing all users.

static authenticated_users ()

Factory method for a member representing all authenticated users.

Return type *string*

Returns A member string representing all authenticated users.

classmethod from_api_repr (*resource*)

Create a policy from the resource returned from the API.

Parameters **resource** (*dict*) – resource returned from the `getIamPolicy` API.

Return type *Policy*

Returns the parsed policy

to_api_repr ()

Construct a Policy resource.

Return type *dict*

Returns a resource to be passed to the `setIamPolicy` API.

Using the API

27.1 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 `GOOGLE_CLOUD_PROJECT` environment variables, create an instance of *Client*.

```
>>> from google.cloud import bigquery
>>> client = bigquery.Client()
```

27.2 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 google.cloud import bigquery
>>> client = bigquery.Client(project='PROJECT_ID')
```

27.2.1 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.

27.3 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.

27.3.1 Dataset operations

List datasets for the client's project:

```
datasets, token = client.list_datasets() # API request
while True:
    for dataset in datasets:
        do_something_with(dataset)
    if token is None:
        break
    datasets, token = client.list_datasets(page_token=token) # API request
```

Create a new dataset for the client's project:

```
dataset = client.dataset(DATASET_NAME)
dataset.create() # API request
```

Check for the existence of a dataset:

```
assert not dataset.exists() # API request
dataset.create() # API request
assert dataset.exists() # API request
```

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

```
assert dataset.description == ORIGINAL_DESCRIPTION
dataset.description = LOCALLY_CHANGED_DESCRIPTION
assert dataset.description == LOCALLY_CHANGED_DESCRIPTION
dataset.reload() # API request
assert dataset.description == ORIGINAL_DESCRIPTION
```

Patch metadata for a dataset:

```
ONE_DAY_MS = 24 * 60 * 60 * 1000
assert dataset.description == ORIGINAL_DESCRIPTION
dataset.patch(
    description=PATCHED_DESCRIPTION,
    default_table_expiration_ms=ONE_DAY_MS
) # API request
assert dataset.description == PATCHED_DESCRIPTION
assert dataset.default_table_expiration_ms == ONE_DAY_MS
```

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

```
>>> from google.cloud 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:

```
assert dataset.exists()      # API request
dataset.delete()
assert not dataset.exists()  # API request
```

27.4 Tables

Tables exist within datasets. List tables for the dataset:

```
tables, token = dataset.list_tables()  # API request
assert len(tables) == 0
assert token is None
table = dataset.table(TABLE_NAME)
table.view_query = QUERY
table.create()                        # API request
tables, token = dataset.list_tables()  # API request
assert len(tables) == 1
assert tables[0].name == TABLE_NAME
```

Create a table:

```
table = dataset.table(TABLE_NAME, SCHEMA)
table.create()                        # API request
```

Check for the existence of a table:

```
table = dataset.table(TABLE_NAME, SCHEMA)
assert not table.exists()             # API request
table.create()                        # API request
assert table.exists()                 # API request
```

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

```
assert table.friendly_name == ORIGINAL_FRIENDLY_NAME
assert table.description == ORIGINAL_DESCRIPTION
table.friendly_name = LOCALLY_CHANGED_FRIENDLY_NAME
table.description = LOCALLY_CHANGED_DESCRIPTION
table.reload()                       # API request
assert table.friendly_name == ORIGINAL_FRIENDLY_NAME
assert table.description == ORIGINAL_DESCRIPTION
```

Patch specific properties for a table:

```
assert table.friendly_name == ORIGINAL_FRIENDLY_NAME
assert table.description == ORIGINAL_DESCRIPTION
table.patch(
    friendly_name=PATCHED_FRIENDLY_NAME,
    description=PATCHED_DESCRIPTION,
)                                     # API request
assert table.friendly_name == PATCHED_FRIENDLY_NAME
assert table.description == PATCHED_DESCRIPTION
```

Update all writable metadata for a table

```
assert table.friendly_name == ORIGINAL_FRIENDLY_NAME
assert table.description == ORIGINAL_DESCRIPTION
NEW_SCHEMA = table.schema[:]
```

```

NEW_SCHEMA.append(SchemaField('phone', 'string'))
table.friendly_name = UPDATED_FRIENDLY_NAME
table.description = UPDATED_DESCRIPTION
table.schema = NEW_SCHEMA
table.update()           # API request
assert table.friendly_name == UPDATED_FRIENDLY_NAME
assert table.description == UPDATED_DESCRIPTION
assert table.schema == NEW_SCHEMA

```

Get rows from a table's data:

```

rows, _, token = table.fetch_data()
while True:
    for row in rows:
        do_something(row)
    if token is None:
        break
rows, _, token = table.fetch_data(page_token=token)

```

Insert rows into a table's data:

```

ROWS_TO_INSERT = [
    (u'Phred Phlyntstone', 32),
    (u'Wylma Phlyntstone', 29),
]

table.insert_data(ROWS_TO_INSERT)

```

Upload table data from a file:

```

writer = csv.writer(csv_file)
writer.writerow((b'full_name', b'age'))
writer.writerow((b'Phred Phlyntstone', b'32'))
writer.writerow((b'Wylma Phlyntstone', b'29'))
csv_file.flush()

with open(csv_file.name, 'rb') as readable:
    table.upload_from_file(
        readable, source_format='CSV', skip_leading_rows=1)

```

Delete a table:

```

assert table.exists()           # API request
table.delete()                   # API request
assert not table.exists()       # API request

```

27.5 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:


```

jobs, token = client.list_jobs() # API request
while True:
    for job in jobs:
        do_something_with(job)
    if token is None:
        break
    jobs, token = client.list_jobs(page_token=token) # API request

```

27.5.1 Querying data (synchronous)

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

```

query = client.run_sync_query(LIMITED)
query.timeout_ms = TIMEOUT_MS
query.run() # API request

assert query.complete
assert len(query.rows) == LIMIT
assert [field.name for field in query.schema] == ['name']

```

If the rows returned by the query do not fit into the initial response, then we need to fetch the remaining rows via `fetch_data`:

```

query = client.run_sync_query(LIMITED)
query.timeout_ms = TIMEOUT_MS
query.max_results = PAGE_SIZE
query.run() # API request

assert query.complete
assert query.page_token is not None
assert len(query.rows) == PAGE_SIZE
assert [field.name for field in query.schema] == ['name']

rows = query.rows
token = query.page_token

while True:
    do_something_with(rows)
    if token is None:
        break
    rows, total_count, token = query.fetch_data(
        page_token=token) # API request

```

If the query takes longer than the timeout allowed, `query.complete` will be `False`. In that case, we need to poll the associated job until it is done, and then fetch the results:

```

query = client.run_sync_query(QUERY)
query.timeout_ms = TIMEOUT_MS
query.use_query_cache = False
query.run() # API request

assert not query.complete

job = query.job
job.reload() # API request
retry_count = 0

```

```

while retry_count < 10 and job.state != u'DONE':
    time.sleep(1.5**retry_count)      # exponential backoff
    retry_count += 1
    job.reload()                       # API request

assert job.state == u'DONE'

rows, total_count, token = query.fetch_data() # API request
while True:
    do_something_with(rows)
    if token is None:
        break
    rows, total_count, token = query.fetch_data(
        page_token=token) # API request

```

27.5.2 Querying data (asynchronous)

Background a query, loading the results into a table:

```

>>> from google.cloud 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
>>> job.write_disposition= 'truncate'
>>> job.name
'fullname-age-query-job'
>>> job.job_type
'query'
>>> job.created
None
>>> job.state
None

```

Note:

- `google.cloud.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 != 'DONE':
...     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>)

```

Retrieve the results:

```

>>> results = job.results()
>>> rows, total_count, token = query.fetch_data() # API request
>>> while True:
...     do_something_with(rows)
...     if token is None:
...         break
...     rows, total_count, token = query.fetch_data(
...         page_token=token) # API request

```

27.5.3 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 google.cloud import bigquery
>>> from google.cloud.bigquery import SchemaField
>>> client = bigquery.Client()
>>> table = dataset.table(name='person_ages')
>>> table.schema = [
...     SchemaField('full_name', 'STRING', mode='required'),
...     SchemaField('age', 'INTEGER', 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:

- `google.cloud.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 != 'DONE':
...     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>)
```

27.5.4 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 google.cloud 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:

- `google.cloud.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 != 'DONE':
...     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>)

```

27.5.5 Copy tables (async)

First, create the job locally:

```

>>> from google.cloud 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:

- `google.cloud.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 != 'DONE':
...     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 `google.cloud.bigquery.client.Client` (*project=None, credentials=None, http=None*)
 Bases: `google.cloud.client.JSONClient`

Client to bundle configuration needed for API requests.

Parameters

- **project** (*str*) – 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** (*str*) – Name of the job.
- **destination** (`google.cloud.bigquery.table.Table`) – Table into which data is to be copied.
- **sources** (sequence of `google.cloud.bigquery.table.Table`) – tables to be copied.

Return type `google.cloud.bigquery.job.CopyJob`

Returns a new `CopyJob` instance

dataset (*dataset_name*)

Construct a dataset bound to this client.

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

Return type `google.cloud.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** (*str*) – Name of the job.
- **source** (*google.cloud.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 *google.cloud.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

Return type One of: *google.cloud.bigquery.job.LoadTableFromStorageJob*,
google.cloud.bigquery.job.CopyJob, *google.cloud.bigquery.job.ExtractTableToStorageJob*,
google.cloud.bigquery.job.QueryJob, *google.cloud.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** (*str*) – 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 *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** (*str*) – 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** (*str*) – 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`).

list_projects (*max_results=None, page_token=None*)

List projects for the project associated with this client.

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

Parameters

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

Return type tuple, (list, str)

Returns list of *Project*, plus a “next page token” string: if the token is not `None`, indicates that more projects can be retrieved with another call (pass 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** (*str*) – Name of the job.
- **destination** (*google.cloud.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 *google.cloud.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** (*str*) – Name of the job.
- **query** (*str*) – SQL query to be executed

Return type *google.cloud.bigquery.job.QueryJob*

Returns a new *QueryJob* instance

run_sync_query (*query*)

Run a SQL query synchronously.

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

Return type `google.cloud.bigquery.query.QueryResults`

Returns a new `QueryResults` instance

class `google.cloud.bigquery.client.Project` (*project_id, numeric_id, friendly_name*)

Bases: `object`

Wrapper for resource describing a BigQuery project.

Parameters

- **project_id** (*str*) – Opaque ID of the project
- **numeric_id** (*int*) – Numeric ID of the project
- **friendly_name** (*str*) – Display name of the project

classmethod `from_api_repr` (*resource*)

Factory: construct an instance from a resource dict.

28.1 Connection

Create / interact with Google Cloud BigQuery connections.

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

Bases: `google.cloud.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 `google.cloud.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 `google.cloud.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** (`google.cloud.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** (*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** (*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** (*Client* or *NoneType*) – the client to use. If not passed, falls back to the `client` stored on the current dataset.

Return type `bool`

Returns Boolean indicating existence of the 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** (*google.cloud.bigquery.client.Client*) – Client which holds credentials and project configuration for the dataset.

Return type *google.cloud.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 *google.cloud.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** (*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** (*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 `google.cloud.bigquery.table.SchemaField`) – The table's schema

Return type `google.cloud.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** (*Client* or `NoneType`) – the client to use. If not passed, falls back to the `client` stored on the current dataset.

Define API Jobs.

class `google.cloud.bigquery.job.Compression` (*name*)
 Bases: `google.cloud.bigquery._helpers._EnumProperty`
 Pseudo-enum for compression properties.

class `google.cloud.bigquery.job.CopyJob` (*name, destination, sources, client*)
 Bases: `google.cloud.bigquery.job._AsyncJob`
 Asynchronous job: copy data into a table from other tables.

Parameters

- **name** (*string*) – the name of the job
- **destination** (*google.cloud.bigquery.table.Table*) – Table into which data is to be loaded.
- **sources** (list of *google.cloud.bigquery.table.Table*) – Table into which data is to be loaded.
- **client** (*google.cloud.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** (*google.cloud.bigquery.client.Client*) – Client which holds credentials and project configuration for the dataset.

Return type *google.cloud.bigquery.job.CopyJob*

Returns Job parsed from *resource*.

`write_disposition`

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

class `google.cloud.bigquery.job.CreateDisposition` (*name*)
 Bases: `google.cloud.bigquery._helpers._EnumProperty`
 Pseudo-enum for `create_disposition` properties.

class `google.cloud.bigquery.job.DestinationFormat` (*name*)
Bases: `google.cloud.bigquery._helpers._EnumProperty`
Pseudo-enum for `destination_format` properties.

class `google.cloud.bigquery.job.Encoding` (*name*)
Bases: `google.cloud.bigquery._helpers._EnumProperty`
Pseudo-enum for encoding properties.

class `google.cloud.bigquery.job.ExtractTableToStorageJob` (*name*, *source*, *destination_uris*, *client*)
Bases: `google.cloud.bigquery.job._AsyncJob`
Asynchronous job: extract data from a table into Cloud Storage.

Parameters

- **name** (*string*) – the name of the job
- **source** (`google.cloud.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** (`google.cloud.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** (`google.cloud.bigquery.client.Client`) – Client which holds credentials and project configuration for the dataset.

Return type `google.cloud.bigquery.job.ExtractTableToStorageJob`

Returns Job parsed from `resource`.

`print_header`

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

class `google.cloud.bigquery.job.LoadTableFromStorageJob` (*name*, *destination*, *source_uris*, *client*, *schema=()*)

Bases: `google.cloud.bigquery.job._AsyncJob`

Asynchronous job for loading data into a table from CloudStorage.

Parameters

- **name** (*string*) – the name of the job

- **destination** (*google.cloud.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** (*google.cloud.bigquery.client.Client*) – A client which holds credentials and project configuration for the dataset (which requires a project).
- **schema** (list of *google.cloud.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** (*google.cloud.bigquery.client.Client*) – Client which holds credentials and project configuration for the dataset.

Return type *google.cloud.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 `google.cloud.bigquery.job.QueryJob` (*name*, *query*, *client*, *udf_resources*=())

Bases: `google.cloud.bigquery.job._AsyncJob`

Asynchronous job: query tables.

Parameters

- **name** (*string*) – the name of the job
- **query** (*string*) – SQL query string
- **client** (*google.cloud.bigquery.client.Client*) – A client which holds credentials and project configuration for the dataset (which requires a project).
- **udf_resources** (*tuple*) – An iterable of `google.cloud.bigquery._helpers.UDFResource` (empty by default)

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** (*google.cloud.bigquery.client.Client*) – Client which holds credentials and project configuration for the dataset.

Return type `google.cloud.bigquery.job.RunAsyncQueryJob`

Returns Job parsed from `resource`.

maximum_billing_tier

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

maximum_bytes_billed

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

priority

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

results ()

Construct a `QueryResults` instance, bound to this job.

Return type *QueryResults*

Returns results instance

use_legacy_sql

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

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 `google.cloud.bigquery.job.QueryPriority` (*name*)
 Bases: `google.cloud.bigquery._helpers._EnumProperty`
 Pseudo-enum for `QueryJob.priority` property.

class `google.cloud.bigquery.job.SourceFormat` (*name*)
 Bases: `google.cloud.bigquery._helpers._EnumProperty`
 Pseudo-enum for `source_format` properties.

class `google.cloud.bigquery.job.WriteDisposition` (*name*)
 Bases: `google.cloud.bigquery._helpers._EnumProperty`
 Pseudo-enum for `write_disposition` properties.

Tables

Define API Datasets.

class `google.cloud.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** (*str*) – the name of the table
- **dataset** (`google.cloud.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** (`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 `str`

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** (`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 str, or NoneType

Returns The description as set by the user, or None (the default).

etag

ETag for the table resource.

Return type str, 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** (*Client* or NoneType) – the client to use. If not passed, falls back to the `client` stored on the current dataset.

Return type bool

Returns Boolean indicating existence of the table.

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** (str or NoneType) – token representing a cursor into the table's rows.
- **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 str, 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** (*google.cloud.bigquery.dataset.Dataset*) – The dataset containing the table.

Return type *google.cloud.bigquery.table.Table*

Returns Table parsed from `resource`.

insert_data (*rows*, *row_ids=None*, *skip_invalid_rows=None*, *ignore_unknown_values=None*, *template_suffix=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?
- **template_suffix** (str or `NoneType`) – treat name as a template table and provide a suffix. BigQuery will create the table `<name> + <template_suffix>` based on the schema of the template table. See: <https://cloud.google.com/bigquery/streaming-data-into-bigquery#template-tables>
- **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.

list_partitions (*client=None*)

List the partitions in a table.

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

Return type list

Returns a list of time partitions

location

Location in which the table is hosted.

Return type str, 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).

partition_expiration

Expiration time in ms for a partition :rtype: int, or NoneType :returns: Returns the time in ms for partition expiration

partitioning_type

Time partitioning of the table. :rtype: str, or NoneType :returns: Returns type if the table is partitioned, None otherwise.

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** (*Client* or `NoneType`) – the client to use. If not passed, falls back to the `client` stored on the current dataset.
- **friendly_name** (`str` or `NoneType`) – point in time at which the table expires.
- **description** (`str` or `NoneType`) – point in time at which the table expires.
- **location** (`str` 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** (*str*) – 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 `str`

Returns the path based on project and dataset name.

project

Project bound to the table.

Return type `str`

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** (*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 str, or *NoneType*

Returns the URL (None until set from the server).

table_id

ID for the table resource.

Return type str, 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 str, 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** (*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 opened in binary mode for reading.
- **source_format** (*str*) – one of 'CSV' or 'NEWLINE_DELIMITED_JSON'. job configuration option; see `google.cloud.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 `google.cloud.bigquery.job.LoadJob()`.
- **allow_quoted_newlines** (*boolean*) – job configuration option; see `google.cloud.bigquery.job.LoadJob()`.
- **create_disposition** (*str*) – job configuration option; see `google.cloud.bigquery.job.LoadJob()`.
- **encoding** (*str*) – job configuration option; see `google.cloud.bigquery.job.LoadJob()`.
- **field_delimiter** (*str*) – job configuration option; see `google.cloud.bigquery.job.LoadJob()`.
- **ignore_unknown_values** (*boolean*) – job configuration option; see `google.cloud.bigquery.job.LoadJob()`.
- **max_bad_records** (*integer*) – job configuration option; see `google.cloud.bigquery.job.LoadJob()`.
- **quote_character** (*str*) – job configuration option; see `google.cloud.bigquery.job.LoadJob()`.
- **skip_leading_rows** (*integer*) – job configuration option; see `google.cloud.bigquery.job.LoadJob()`.
- **write_disposition** (*str*) – job configuration option; see `google.cloud.bigquery.job.LoadJob()`.
- **client** (*Client* or *NoneType*) – Optional. The client to use. If not passed, falls back to the `client` stored on the current dataset.

Return type `google.cloud.bigquery.jobs.LoadTableFromStorageJob`

Returns the job instance used to load the data (e.g., for querying status). Note that the job is already started: do not call `job.begin()`.

Raises `ValueError` if `size` is not passed in and can not be determined, or if the `file_obj` can be detected to be a file opened in text mode.

view_query

SQL query defining the table as a view.

Return type `str`, or `NoneType`

Returns The query as set by the user, or `None` (the default).

Query

Define API Queries.

class `google.cloud.bigquery.query.QueryResults` (*query*, *client*, *udf_resources*=())
 Bases: `object`

Synchronous job: query tables.

Parameters

- **query** (*string*) – SQL query string
- **client** (`google.cloud.bigquery.client.Client`) – A client which holds credentials and project configuration for the dataset (which requires a project).
- **udf_resources** (*tuple*) – An iterable of `google.cloud.bigquery.job.UDFResource` (empty by default)

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>

dry_run

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

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** (*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.

classmethod from_query_job (*job*)

Factory: construct from an existing job.

Parameters *job* (*QueryJob*) – existing job

Return type *QueryResults*

Returns the instance, bound to the job

job

Job instance used to run the query.

Return type *google.cloud.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** (*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_legacy_sql

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

use_query_cache

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

Schemas

Schemas for BigQuery tables / queries.

```
class google.cloud.bigquery.schema.SchemaField(name, field_type, mode='NULLABLE', description=None, fields=None)
```

Bases: `object`

Describe a single field within a table schema.

Parameters

- **name** (*str*) – the name of the field.
- **field_type** (*str*) – the type of the field (one of 'STRING', 'INTEGER', 'FLOAT', 'BOOLEAN', 'TIMESTAMP' or 'RECORD').
- **mode** (*str*) – the type of the field (one of 'NULLABLE', 'REQUIRED', or 'REPEATED').
- **description** (*str*) – optional description for the field.
- **fields** (list of *SchemaField*, or None) – subfields (requires `field_type` of 'RECORD').

Using the API

API requests are sent to the [Google Cloud Bigtable API](#) via RPC over HTTP/2. In order to support this, we'll rely on [gRPC](#). We are working with the [gRPC](#) team to rapidly make the install story more user-friendly.

Get started by learning about the `Client` on the [Base for Everything](#) page.

In the hierarchy of API concepts

- a `Client` owns a `Cluster` `<google.cloud.bigtable.instance.Instance`
- a `Cluster` `<google.cloud.bigtable.instance.Instance` owns a `Table`
- a `Table` owns a `ColumnFamily`
- a `Table` owns a `Row` (and all the cells in the row)

Base for Everything

To use the API, the `Client` class defines a high-level interface which handles authorization and creating other objects:

```
from google.cloud.bigtable.client import Client
client = Client()
```

35.1 Long-lived Defaults

When creating a `Client`, the `user_agent` argument has sensible a default (`DEFAULT_USER_AGENT`). However, you may over-ride it and the value will be used throughout all API requests made with the `client` you create.

35.2 Configuration

- For an overview of authentication in `google-cloud-python`, see [Authentication](#).
- In addition to any authentication configuration, you can also set the `GOOGLE_CLOUD_PROJECT` environment variable for the Google Cloud Console project you'd like to interact with. If your code is running in Google App Engine or Google Compute Engine the project will be detected automatically. (Setting this environment variable is not required, you may instead pass the `project` explicitly when constructing a `Client`).
- After configuring your environment, create a `Client`

```
>>> from google.cloud import bigtable
>>> client = bigtable.Client()
```

or pass in credentials and project explicitly

```
>>> from google.cloud import bigtable
>>> client = bigtable.Client(project='my-project', credentials=creds)
```

Tip: Be sure to use the **Project ID**, not the **Project Number**.

35.3 Admin API Access

If you'll be using your client to make `Instance Admin` and `Table Admin` API requests, you'll need to pass the `admin` argument:

```
client = bigtable.Client(admin=True)
```

35.4 Read-Only Mode

If on the other hand, you only have (or want) read access to the data, you can pass the `read_only` argument:

```
client = bigtable.Client(read_only=True)
```

This will ensure that the `READ_ONLY_SCOPE` is used for API requests (so any accidental requests that would modify data will fail).

35.5 Next Step

After a `Client`, the next highest-level object is a `Instance`. You'll need one before you can interact with tables or data.

Head next to learn about the [Instance Admin API](#).

Instance Admin API

After creating a `Client`, you can interact with individual instances for a project.

36.1 List Instances

If you want a comprehensive list of all existing instances, make a `ListInstances` API request with `Client.list_instances()`:

```
instances = client.list_instances()
```

36.2 Instance Factory

To create a `Instance` object:

```
instance = client.instance(instance_id, location_id,  
                           display_name=display_name)
```

- `location_id` is the ID of the location in which the instance's cluster will be hosted, e.g. 'us-centrall1-c'. `location_id` is required for instances which do not already exist.
- `display_name` is optional. When not provided, `display_name` defaults to the `instance_id` value.

You can also use `Client.instance()` to create a local wrapper for instances that have already been created with the API, or through the web console:

```
instance = client.instance(existing_instance_id)  
instance.reload()
```

36.3 Create a new Instance

After creating the instance object, make a `CreateInstance` API request with `create()`:

```
instance.display_name = 'My very own instance'  
instance.create()
```

36.4 Check on Current Operation

Note: When modifying a instance (via a `CreateInstance` request), the Bigtable API will return a long-running operation and a corresponding `Operation` object will be returned by `create()`.

You can check if a long-running operation (for a `create()` has finished by making a `GetOperation` request with `Operation.finished()`:

```
>>> operation = instance.create()
>>> operation.finished()
True
```

Note: Once an `Operation` object has returned `True` from `finished()`, the object should not be re-used. Subsequent calls to `finished()` will result in a `ValueError`.

36.5 Get metadata for an existing Instance

After creating the instance object, make a `GetInstance` API request with `reload()`:

```
instance.reload()
```

This will load `display_name` for the existing `instance` object.

36.6 Update an existing Instance

After creating the instance object, make an `UpdateInstance` API request with `update()`:

```
client.display_name = 'New display_name'
instance.update()
```

36.7 Delete an existing Instance

Make a `DeleteInstance` API request with `delete()`:

```
instance.delete()
```

36.8 Next Step

Now we go down the hierarchy from `Instance` to a `Table`.

Head next to learn about the [Table Admin API](#).

Table Admin API

After creating a `Instance`, you can interact with individual tables, groups of tables or column families within a table.

37.1 List Tables

If you want a comprehensive list of all existing tables in a instance, make a `ListTables` API request with `Instance.list_tables()`:

```
>>> instance.list_tables()
[<google.cloud.bigtable.table.Table at 0x7ff6a1de8f50>,
 <google.cloud.bigtable.table.Table at 0x7ff6a1de8350>]
```

37.2 Table Factory

To create a `Table` object:

```
table = instance.table(table_id)
```

Even if this `Table` already has been created with the API, you'll want this object to use as a parent of a `ColumnFamily` or `Row`.

37.3 Create a new Table

After creating the table object, make a `CreateTable` API request with `create()`:

```
table.create()
```

If you would to initially split the table into several tablets (Tablets are similar to HBase regions):

```
table.create(initial_split_keys=['s1', 's2'])
```

37.4 Delete an existing Table

Make a `DeleteTable` API request with `delete()`:

```
table.delete()
```

37.5 List Column Families in a Table

Though there is no **official** method for retrieving [column families](#) associated with a table, the `GetTable` API method returns a table object with the names of the column families.

To retrieve the list of column families use `list_column_families()`:

```
column_families = table.list_column_families()
```

37.6 Column Family Factory

To create a `ColumnFamily` object:

```
column_family = table.column_family(column_family_id)
```

There is no real reason to use this factory unless you intend to create or delete a column family.

In addition, you can specify an optional `gc_rule` (a `GarbageCollectionRule` or similar):

```
column_family = table.column_family(column_family_id,  
                                     gc_rule=gc_rule)
```

This rule helps the backend determine when and how to clean up old cells in the column family.

See [Column Families](#) for more information about `GarbageCollectionRule` and related classes.

37.7 Create a new Column Family

After creating the column family object, make a `CreateColumnFamily` API request with `ColumnFamily.create()`

```
column_family.create()
```

37.8 Delete an existing Column Family

Make a `DeleteColumnFamily` API request with `ColumnFamily.delete()`

```
column_family.delete()
```

37.9 Update an existing Column Family

Make an `UpdateColumnFamily` API request with `ColumnFamily.delete()`

```
column_family.update()
```


37.10 Next Step

Now we go down the final step of the hierarchy from `Table` to `Row` as well as streaming data directly via a `Table`. Head next to learn about the [Data API](#).

After creating a `Table` and some column families, you are ready to store and retrieve data.

38.1 Cells vs. Columns vs. Column Families

- As explained in the [table overview](#), tables can have many column families.
- As described below, a table can also have many rows which are specified by row keys.
- Within a row, data is stored in a cell. A cell simply has a value (as bytes) and a timestamp. The number of cells in each row can be different, depending on what was stored in each row.
- Each cell lies in a column (**not** a column family). A column is really just a more **specific** modifier within a column family. A column can be present in every column family, in only one or anywhere in between.
- Within a column family there can be many columns. For example within the column family `foo` we could have columns `bar` and `baz`. These would typically be represented as `foo:bar` and `foo:baz`.

38.2 Modifying Data

Since data is stored in cells, which are stored in rows, we use the metaphor of a **row** in classes that are used to modify (write, update, delete) data in a `Table`.

38.2.1 Direct vs. Conditional vs. Append

There are three ways to modify data in a table, described by the `MutateRow`, `CheckAndMutateRow` and `ReadModifyWriteRow` API methods.

- The **direct** way is via `MutateRow` which involves simply adding, overwriting or deleting cells. The `DirectRow` class handles direct mutations.
- The **conditional** way is via `CheckAndMutateRow`. This method first checks if some filter is matched in a given row, then applies one of two sets of mutations, depending on if a match occurred or not. (These mutation sets are called the “true mutations” and “false mutations”.) The `ConditionalRow` class handles conditional mutations.
- The **append** way is via `ReadModifyWriteRow`. This simply appends (as bytes) or increments (as an integer) data in a presumed existing cell in a row. The `AppendRow` class handles append mutations.

38.2.2 Row Factory

A single factory can be used to create any of the three row types. To create a `DirectRow`:

```
row = table.row(row_key)
```

Unlike the previous string values we've used before, the row key must be bytes.

To create a `ConditionalRow`, first create a `RowFilter` and then

```
cond_row = table.row(row_key, filter_=filter_)
```

To create an `AppendRow`

```
append_row = table.row(row_key, append=True)
```

38.2.3 Building Up Mutations

In all three cases, a set of mutations (or two sets) are built up on a row before they are sent of in a batch via

```
row.commit()
```

38.2.4 Direct Mutations

Direct mutations can be added via one of four methods

- `set_cell()` allows a single value to be written to a column

```
row.set_cell(column_family_id, column, value,  
            timestamp=timestamp)
```

If the `timestamp` is omitted, the current time on the Google Cloud Bigtable server will be used when the cell is stored.

The value can either be bytes or an integer (which will be converted to bytes as a signed 64-bit integer).

- `delete_cell()` deletes all cells (i.e. for all timestamps) in a given column

```
row.delete_cell(column_family_id, column)
```

Remember, this only happens in the `row` we are using.

If we only want to delete cells from a limited range of time, a `TimestampRange` can be used

```
row.delete_cell(column_family_id, column,  
              time_range=time_range)
```

- `delete_cells()` does the same thing as `delete_cell()` but accepts a list of columns in a column family rather than a single one.

```
row.delete_cells(column_family_id, [column1, column2],  
                time_range=time_range)
```

In addition, if we want to delete cells from every column in a column family, the special `ALL_COLUMNS` value can be used

```
row.delete_cells(column_family_id, row.ALL_COLUMNS,  
                time_range=time_range)
```

- `delete()` will delete the entire row

```
row.delete()
```

38.2.5 Conditional Mutations

Making **conditional** modifications is essentially identical to **direct** modifications: it uses the exact same methods to accumulate mutations.

However, each mutation added must specify a `state`: will the mutation be applied if the filter matches or if it fails to match.

For example:

```
cond_row.set_cell(column_family_id, column, value,
                  timestamp=timestamp, state=True)
```

will add to the set of true mutations.

38.2.6 Append Mutations

Append mutations can be added via one of two methods

- `append_cell_value()` appends a bytes value to an existing cell:

```
append_row.append_cell_value(column_family_id, column, bytes_value)
```

- `increment_cell_value()` increments an integer value in an existing cell:

```
append_row.increment_cell_value(column_family_id, column, int_value)
```

Since only bytes are stored in a cell, the cell value is decoded as a signed 64-bit integer before being incremented. (This happens on the Google Cloud Bigtable server, not in the library.)

Notice that no timestamp was specified. This is because **append** mutations operate on the latest value of the specified column.

If there are no cells in the specified column, then the empty string (bytes case) or zero (integer case) are the assumed values.

38.2.7 Starting Fresh

If accumulated mutations need to be dropped, use

```
row.clear()
```

38.3 Reading Data

38.3.1 Read Single Row from a Table

To make a `ReadRows` API request for a single row key, use `Table.read_row()`:

```

>>> row_data = table.read_row(row_key)
>>> row_data.cells
{
  u'fam1': {
    b'col1': [
      <google.cloud.bigtable.row_data.Cell at 0x7f80d150ef10>,
      <google.cloud.bigtable.row_data.Cell at 0x7f80d150ef10>,
    ],
    b'col2': [
      <google.cloud.bigtable.row_data.Cell at 0x7f80d150ef10>,
    ],
  },
  u'fam2': {
    b'col3': [
      <google.cloud.bigtable.row_data.Cell at 0x7f80d150ef10>,
      <google.cloud.bigtable.row_data.Cell at 0x7f80d150ef10>,
      <google.cloud.bigtable.row_data.Cell at 0x7f80d150ef10>,
    ],
  },
}
>>> cell = row_data.cells[u'fam1'][b'col1'][0]
>>> cell
<google.cloud.bigtable.row_data.Cell at 0x7f80d150ef10>
>>> cell.value
b'vall'
>>> cell.timestamp
datetime.datetime(2016, 2, 27, 3, 41, 18, 122823, tzinfo=<UTC>)

```

Rather than returning a `DirectRow` or similar class, this method returns a `PartialRowData` instance. This class is used for reading and parsing data rather than for modifying data (as `DirectRow` is).

A filter can also be applied to the results:

```
row_data = table.read_row(row_key, filter_=filter_val)
```

The allowable `filter_` values are the same as those used for a `ConditionalRow`. For more information, see the `Table.read_row()` documentation.

38.3.2 Stream Many Rows from a Table

To make a `ReadRows` API request for a stream of rows, use `Table.read_rows()`:

```
row_data = table.read_rows()
```

Using gRPC over HTTP/2, a continual stream of responses will be delivered. In particular

- `consume_next()` pulls the next result from the stream, parses it and stores it on the `PartialRowsData` instance
- `consume_all()` pulls results from the stream until there are no more
- `cancel()` closes the stream

See the `PartialRowsData` documentation for more information.

As with `Table.read_row()`, an optional `filter_` can be applied. In addition a `start_key` and / or `end_key` can be supplied for the stream, a `limit` can be set and a boolean `allow_row_interleaving` can be specified to allow faster streamed results at the potential cost of non-sequential reads.

See the `Table.read_rows()` documentation for more information on the optional arguments.

38.3.3 Sample Keys in a Table

Make a `SampleRowKeys` API request with `Table.sample_row_keys()`:

```
keys_iterator = table.sample_row_keys()
```

The returned row keys will delimit contiguous sections of the table of approximately equal size, which can be used to break up the data for distributed tasks like mapreduces.

As with `Table.read_rows()`, the returned `keys_iterator` is connected to a cancellable HTTP/2 stream.

The next key in the result can be accessed via

```
next_key = keys_iterator.next()
```

or all keys can be iterated over via

```
for curr_key in keys_iterator:  
    do_something(curr_key)
```

Just as with reading, the stream can be canceled:

```
keys_iterator.cancel()
```

Client

Instance

Cluster

Table

Column Families

When creating a `ColumnFamily`, it is possible to set garbage collection rules for expired data.

By setting a rule, cells in the table matching the rule will be deleted during periodic garbage collection (which executes opportunistically in the background).

The types `MaxAgeGCRule`, `MaxVersionsGCRule`, `GarbageCollectionRuleUnion` and `GarbageCollectionRuleIntersection` can all be used as the optional `gc_rule` argument in the `ColumnFamily` constructor. This value is then used in the `create()` and `update()` methods.

These rules can be nested arbitrarily, with a `MaxAgeGCRule` or `MaxVersionsGCRule` at the lowest level of the nesting:

```
import datetime

max_age = datetime.timedelta(days=3)
rule1 = MaxAgeGCRule(max_age)
rule2 = MaxVersionsGCRule(1)

# Make a composite that matches anything older than 3 days **AND**
# with more than 1 version.
rule3 = GarbageCollectionIntersection(rules=[rule1, rule2])

# Make another composite that matches our previous intersection
# **OR** anything that has more than 3 versions.
rule4 = GarbageCollectionRule(max_num_versions=3)
rule5 = GarbageCollectionUnion(rules=[rule3, rule4])
```

Bigtable Row

Bigtable Row Filters

It is possible to use a `RowFilter` when adding mutations to a `ConditionalRow` and when reading row data with `read_row()` `read_rows()`.

As laid out in the `RowFilter` definition, the following basic filters are provided:

- `SinkFilter`
- `PassAllFilter`
- `BlockAllFilter`
- `RowKeyRegexFilter`
- `RowSampleFilter`
- `FamilyNameRegexFilter`
- `ColumnQualifierRegexFilter`
- `TimestampRangeFilter`
- `ColumnRangeFilter`
- `ValueRegexFilter`
- `ValueRangeFilter`
- `CellsRowOffsetFilter`
- `CellsRowLimitFilter`
- `CellsColumnLimitFilter`
- `StripValueTransformerFilter`
- `ApplyLabelFilter`

In addition, these filters can be combined into composite filters with

- `RowFilterChain`
- `RowFilterUnion`
- `ConditionalRowFilter`

These rules can be nested arbitrarily, with a basic filter at the lowest level. For example:

```
# Filter in a specified column (matching any column family).
coll_filter = ColumnQualifierRegexFilter(b'columnbia')

# Create a filter to label results.
```

```
label1 = u'label-red'
label1_filter = ApplyLabelFilter(label1)

# Combine the filters to label all the cells in columbia.
chain1 = RowFilterChain(filters=[col1_filter, label1_filter])

# Create a similar filter to label cells blue.
col2_filter = ColumnQualifierRegexFilter(b'columnseeya')
label2 = u'label-blue'
label2_filter = ApplyLabelFilter(label2)
chain2 = RowFilterChain(filters=[col2_filter, label2_filter])

# Bring our two labeled columns together.
row_filter = RowFilterUnion(filters=[chain1, chain2])
```

Row Data

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.

Warning: Alpha

The `projects.create()` API method is in the Alpha stage. It might be changed in backward-incompatible ways and is not recommended for production use. It is not subject to any SLA or deprecation policy. Access to this feature is currently invite-only. For an invitation, contact our sales team at <https://cloud.google.com/contact>.

Here's a quick example of the full life-cycle:

```
>>> from google.cloud 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()
```

47.1 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, `google-cloud-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 google.cloud import resource_manager
>>> client = resource_manager.Client()
```

Client

A Client for interacting with the Resource Manager API.

class `google.cloud.resource_manager.client.Client` (*credentials=None, http=None*)
 Bases: `google.cloud.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 google.cloud 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 google.cloud 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 google.cloud 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*)

Create 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.

48.1 Connection

Create / interact with Google Cloud Resource Manager connections.

class `google.cloud.resource_manager.connection.Connection` (*credentials=None*,
http=None)

Bases: `google.cloud.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 `google.cloud.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 google.cloud 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** (*google.cloud.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 (*google.cloud.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** (`google.cloud.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** (`google.cloud.resource_manager.client.Client` or `NoneType`) – the client to use. If not passed, falls back to the client stored on the current project.

Return type `bool`

Returns Boolean indicating existence of the 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** (`google.cloud.resource_manager.client.Client`) – The Client used with this project.

Return type `google.cloud.resource_manager.project.Project`

Returns The project created.

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** (`google.cloud.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.

undelele (*client=None, reload_data=False*)

API call: undelele the project via a POST request.

See <https://cloud.google.com/resource-manager/reference/rest/v1beta1/projects/undelele>

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** (*google.cloud.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** (*google.cloud.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

50.1 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 `google-cloud-python`, see [Authentication](#).

Assuming your environment is set up as described in that document, create an instance of `Client`.

```
>>> from google.cloud import dns
>>> client = dns.Client()
```

50.2 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 google.cloud import dns
>>> client = dns.Client(project='PROJECT_ID')
```

50.3 Project Quotas

Query the quotas for a given project:

```
>>> from google.cloud 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
```

50.3.1 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.

50.4 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 google.cloud import dns
>>> client = dns.Client(project='PROJECT_ID')
>>> zone = client.zone('acme-co', 'example.com',
...                   description='Acme Company zone')

>>> zone.exists() # API request
False
>>> zone.create() # API request
>>> zone.exists() # API request
True
```

List the zones for a given project:

```
>>> from google.cloud import dns
>>> client = dns.Client(project='PROJECT_ID')
>>> zones = client.list_zones() # API request
>>> [zone.name for zone in zones]
['acme-co']
```

50.5 Resource Record Sets

Each managed zone exposes a read-only set of resource records:

```
>>> from google.cloud import dns
>>> client = dns.Client(project='PROJECT_ID')
>>> zone = client.zone('acme-co', 'example.com')
>>> records, page_token = zone.list_resource_record_sets() # API request
>>> [(record.name, record.record_type, record.ttl, record.rdatas)
...  for record in records]
[('example.com.', 'SOA', 21600, ['ns-cloud1.googlecomains.com dns-admin.google.com 1 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)
```

50.6 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 google.cloud import dns
>>> client = dns.Client(project='PROJECT_ID')
>>> zone = client.zone('acme-co', 'example.com')
>>> TWO_HOURS = 2 * 60 * 60 # seconds
>>> record_set = zone.resource_record_set(
...     'www.example.com.', 'CNAME', TWO_HOURS, ['www1.example.com.'])
>>> changes = zone.changes()
>>> changes.add_record_set(record_set)
>>> changes.create() # API request
>>> while changes.status != 'done':
...     print('Waiting for changes to complete')
...     time.sleep(60) # or whatever interval is appropriate
...     changes.reload() # API request
```

List changes made to the resource record set for a given zone:

```
>>> from google.cloud import dns
>>> client = dns.Client(project='PROJECT_ID')
>>> zone = client.zone('acme-co', 'example.com')
>>> 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 `google.cloud.dns.client.Client` (*project=None, credentials=None, http=None*)
 Bases: `google.cloud.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 `google.cloud.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=None*, *description=None*)

Construct a zone bound to this client.

Parameters

- **name** (*string*) – Name of the zone.
- **dns_name** (*string* or *NoneType*) – DNS name of the zone. If not passed, then calls to `zone.create()` will fail.
- **description** (*string* or *NoneType*) – the description for the zone. If not passed, defaults to the value of ‘`dns_name`’.

Return type *google.cloud.dns.zone.ManagedZone*

Returns a new `ManagedZone` instance.

51.1 Connection

Create / interact with Google Cloud DNS connections.

class `google.cloud.dns.connection.Connection` (*credentials=None*, *http=None*)

Bases: *google.cloud.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.cloudndns.readwrite`’,)

The scopes required for authenticating as a Cloud DNS consumer.

Managed Zones

Define API ManagedZones.

```
class google.cloud.dns.zone.ManagedZone (name, dns_name=None, client=None, description=None)
```

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 or `NoneType`) – the DNS name of the zone. If not passed, then calls to `create()` will fail.
- **client** (*google.cloud.dns.client.Client*) – A client which holds credentials and project configuration for the zone (which requires a project).
- **description** (string or `NoneType`) – the description for the zone. If not passed, defaults to the value of 'dns_name'.

changes ()

Construct a change set bound to this zone.

Return type `google.cloud.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** (*google.cloud.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` (`google.cloud.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` (`google.cloud.dns.client.Client` or `NoneType`) – the client to use. If not passed, falls back to the `client` stored on the current zone.

Return type `bool`

Returns Boolean indicating existence of the managed 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** (`google.cloud.dns.client.Client`) – Client which holds credentials and project configuration for the zone.

Return type `google.cloud.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** (`google.cloud.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 `google.cloud.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** (*google.cloud.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 *google.cloud.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** (*google.cloud.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 *google.cloud.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 google.cloud.dns.resource_record_set.ResourceRecordSet (name, record_type, ttl,
                                                             rrdatas, zone)
```

Bases: `object`

ResourceRecordSets are DNS resource records.

RRS are owned by a `google.cloud.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** (`google.cloud.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** (`google.cloud.dns.zone.ManagedZone`) – A zone which holds one or more record sets.

Return type `google.cloud.dns.zone.ResourceRecordSet`

Returns RRS parsed from `resource`.

Change Sets

Define API ResourceRecordSets.

class `google.cloud.dns.changes.Changes` (*zone*)

Bases: `object`

Changes are bundled additions / deletions of DNS resource records.

Changes are owned by a `google.cloud.dns.zone.ManagedZone` instance.

See: <https://cloud.google.com/dns/api/v1/changes>

Parameters `zone` (`google.cloud.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` (`google.cloud.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 `google.cloud.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` (`google.cloud.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` (`google.cloud.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 `google.cloud.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** (*google.cloud.dns.client.Client* or `NoneType`) – the client to use. If not passed, falls back to the `client` stored on the current zone.

Return type `bool`

Returns Boolean indicating existence of the changes.

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** (*google.cloud.dns.zone.ManagedZone*) – A zone which holds zero or more change sets.

Return type *google.cloud.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** (*google.cloud.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

55.1 Authentication and Configuration

- For an overview of authentication in `google-cloud-python`, see [Authentication](#).
- In addition to any authentication configuration, you should also set the `GOOGLE_CLOUD_PROJECT` environment variable for the project you'd like to interact with. If you are Google App Engine or Google Compute Engine this will be detected automatically.
- The library now enables the gRPC transport for the logging API by default, assuming that the required dependencies are installed and importable. To *disable* this transport, set the `GOOGLE_CLOUD_DISABLE_GRPC` environment variable to a non-empty string, e.g.: `$ export GOOGLE_CLOUD_DISABLE_GRPC=true`.
- After configuring your environment, create a `Client`

```
>>> from google.cloud import logging
>>> client = logging.Client()
```

or pass in credentials and project explicitly

```
>>> from google.cloud import logging
>>> client = logging.Client(project='my-project', credentials=creds)
```

55.2 Writing log entries

Write a simple text entry to a logger.

```
>>> from google.cloud import logging
>>> client = logging.Client()
>>> logger = client.logger('log_name')
>>> logger.log_text("A simple entry") # API call
```

Write a dictionary entry to a logger.

```
>>> from google.cloud import logging
>>> client = logging.Client()
>>> logger = client.logger('log_name')
>>> logger.log_struct(
...     message="My second entry",
...     weather="partly cloudy") # API call
```

55.3 Retrieving log entries

Fetch entries for the default project.

```
>>> from google.cloud import logging
>>> client = logging.Client()
>>> entries, token = client.list_entries() # API call
>>> for entry in entries:
...     timestamp = entry.timestamp.isoformat()
...     print('%sZ: %s' %
...           (timestamp, entry.payload))
2016-02-17T20:35:49.031864072Z: A simple entry | None
2016-02-17T20:38:15.944418531Z: None | {'message': 'My second entry', 'weather': 'partly cloudy'}
```

Fetch entries across multiple projects.

```
>>> from google.cloud import logging
>>> client = logging.Client()
>>> entries, token = client.list_entries(
...     project_ids=['one-project', 'another-project']) # API call
```

Filter entries retrieved using the Advanced Logs Filters syntax

```
>>> from google.cloud import logging
>>> client = logging.Client()
>>> FILTER = "log:log_name AND textPayload:simple"
>>> entries, token = client.list_entries(filter=FILTER) # API call
```

Sort entries in descending timestamp order.

```
>>> from google.cloud import logging
>>> client = logging.Client()
>>> entries, token = client.list_entries(order_by=logging.DESCEDING) # API call
```

Retrieve entries in batches of 10, iterating until done.

```
>>> from google.cloud import logging
>>> client = logging.Client()
>>> retrieved = []
>>> token = None
>>> while True:
...     entries, token = client.list_entries(page_size=10, page_token=token) # API call
...     retrieved.extend(entries)
...     if token is None:
...         break
```

Retrieve entries for a single logger, sorting in descending timestamp order:

```
>>> from google.cloud import logging
>>> client = logging.Client()
>>> logger = client.logger('log_name')
>>> entries, token = logger.list_entries(order_by=logging.DESCEDING) # API call
```

55.4 Delete all entries for a logger

```
>>> from google.cloud import logging
>>> client = logging.Client()
```

```
>>> logger = client.logger('log_name')
>>> logger.delete() # API call
```

55.5 Manage log metrics

Metrics are counters of entries which match a given filter. They can be used within Stackdriver Monitoring to create charts and alerts.

Create a metric:

```
>>> from google.cloud import logging
>>> client = logging.Client()
>>> metric = client.metric(
...     "robots", "Robots all up in your server",
...     filter='log:apache-access AND textPayload:robot')
>>> metric.exists() # API call
False
>>> metric.create() # API call
>>> metric.exists() # API call
True
```

List all metrics for a project:

```
>>> from google.cloud import logging
>>> client = logging.Client()
>>> metrics, token = client.list_metrics()
>>> len(metrics)
1
>>> metric = metrics[0]
>>> metric.name
"robots"
```

Refresh local information about a metric:

```
>>> from google.cloud import logging
>>> client = logging.Client()
>>> metric = client.metric("robots")
>>> metric.reload() # API call
>>> metric.description
"Robots all up in your server"
>>> metric.filter_
"log:apache-access AND textPayload:robot"
```

Update a metric:

```
>>> from google.cloud import logging
>>> client = logging.Client()
>>> metric = client.metric("robots")
>>> metric.exists() # API call
True
>>> metric.reload() # API call
>>> metric.description = "Danger, Will Robinson!"
>>> metric.update() # API call
```

Delete a metric:

```
>>> from google.cloud import logging
>>> client = logging.Client()
>>> metric = client.metric("robots")
>>> metric.exists() # API call
True
>>> metric.delete() # API call
>>> metric.exists() # API call
False
```

55.6 Export log entries using sinks

Sinks allow exporting entries which match a given filter to Cloud Storage buckets, BigQuery datasets, or Cloud Pub/Sub topics.

55.6.1 Export to Cloud Storage

Make sure that the storage bucket you want to export logs too has `cloud-logs@google.com` as the owner. See [Set permission for writing exported logs](#).

Add `cloud-logs@google.com` as the owner of `my-bucket-name`:

```
>>> from google.cloud import storage
>>> client = storage.Client()
>>> bucket = client.get_bucket('my-bucket-name')
>>> bucket.acl.reload()
>>> logs_group = bucket.acl.group('cloud-logs@google.com')
>>> logs_group.grant_owner()
>>> bucket.acl.add_entity(logs_group)
>>> bucket.acl.save()
```

55.6.2 Export to BigQuery

To export logs to BigQuery you must log into the Cloud Platform Console and add `cloud-logs@google.com` to a dataset.

See: [Setting permissions for BigQuery](#)

```
>>> from google.cloud import bigquery
>>> from google.cloud.bigquery.dataset import AccessGrant
>>> bigquery_client = bigquery.Client()
>>> dataset = bigquery_client.dataset('my-dataset-name')
>>> dataset.create()
>>> dataset.reload()
>>> grants = dataset.access_grants
>>> grants.append(AccessGrant(
...     'WRITER', 'groupByEmail', 'cloud-logs@google.com'))
>>> dataset.access_grants = grants
>>> dataset.update()
```

55.6.3 Export to Pub/Sub

To export logs to BigQuery you must log into the Cloud Platform Console and add `cloud-logs@google.com` to a topic.

See: [Setting permissions for Pub/Sub](#)

```
>>> from google.cloud import pubsub
>>> client = pubsub.Client()
>>> topic = client.topic('your-topic-name')
>>> policy = topic.get_iam_policy()
>>> policy.owners.add(policy.group('cloud-logs@google.com'))
>>> topic.set_iam_policy(policy)
```

Create a Cloud Storage sink:

```
>>> from google.cloud import logging
>>> client = logging.Client()
>>> sink = client.sink(
...     "robots-storage",
...     'log:apache-access AND textPayload:robot',
...     'storage.googleapis.com/my-bucket-name')
>>> sink.exists() # API call
False
>>> sink.create() # API call
>>> sink.exists() # API call
True
```

Create a BigQuery sink:

```
>>> from google.cloud import logging
>>> client = logging.Client()
>>> sink = client.sink(
...     "robots-bq",
...     'log:apache-access AND textPayload:robot',
...     'bigquery.googleapis.com/projects/projects/my-project/datasets/my-dataset')
>>> sink.exists() # API call
False
>>> sink.create() # API call
>>> sink.exists() # API call
True
```

Create a Cloud Pub/Sub sink:

```
>>> from google.cloud import logging
>>> client = logging.Client()

>>> sink = client.sink(
...     "robots-pubsub",
...     'log:apache-access AND textPayload:robot',
...     'pubsub.googleapis.com/projects/my-project/topics/my-topic')
>>> sink.exists() # API call
False
>>> sink.create() # API call
>>> sink.exists() # API call
True
```

List all sinks for a project:

```
>>> from google.cloud import logging
>>> client = logging.Client()
>>> sinks, token = client.list_sinks()
>>> for sink in sinks:
...     print('%s: %s' % (sink.name, sink.destination))
robots-storage: storage.googleapis.com/my-bucket-name
```

```
robots-bq: bigquery.googleapis.com/projects/my-project/datasets/my-dataset
robots-pubsub: pubsub.googleapis.com/projects/my-project/topics/my-topic
```

Refresh local information about a sink:

```
>>> from google.cloud import logging
>>> client = logging.Client()
>>> sink = client.sink('robots-storage')
>>> sink.filter_ is None
True
>>> sink.reload() # API call
>>> sink.filter_
'log:apache-access AND textPayload:robot'
>>> sink.destination
'storage.googleapis.com/my-bucket-name'
```

Update a sink:

```
>>> from google.cloud import logging
>>> client = logging.Client()
>>> sink = client.sink("robots")
>>> sink.reload() # API call
>>> sink.filter_ = "log:apache-access"
>>> sink.update() # API call
```

Delete a sink:

```
>>> from google.cloud import logging
>>> client = logging.Client()
>>> sink = client.sink(
...     "robots",
...     filter='log:apache-access AND textPayload:robot')
>>> sink.exists() # API call
True
>>> sink.delete() # API call
>>> sink.exists() # API call
False
```

55.7 Integration with Python logging module

It's possible to tie the Python `logging` module directly into Google Cloud Logging. To use it, create a `CloudLoggingHandler` instance from your Logging client.

```
>>> import logging
>>> import google.cloud.logging # Don't conflict with standard logging
>>> from google.cloud.logging.handlers import CloudLoggingHandler
>>> client = google.cloud.logging.Client()
>>> handler = CloudLoggingHandler(client)
>>> cloud_logger = logging.getLogger('cloudLogger')
>>> cloud_logger.setLevel(logging.INFO) # defaults to WARN
>>> cloud_logger.addHandler(handler)
>>> cloud_logger.error('bad news')
```

Note:

This handler by default uses an asynchronous transport that sends log entries on a background thread. However, the API call will still be made in the same process. For other transport options, see the transports section.

All logs will go to a single custom log, which defaults to “python”. The name of the Python logger will be included in the structured log entry under the “python_logger” field. You can change it by providing a name to the handler:

```
>>> handler = CloudLoggingHandler(client, name="mycustomlog")
```

It is also possible to attach the handler to the root Python logger, so that for example a plain `logging.warn` call would be sent to Cloud Logging, as well as any other loggers created. However, you must avoid infinite recursion from the logging calls the client itself makes. A helper method `setup_logging` is provided to configure this automatically:

```
>>> import logging
>>> import google.cloud.logging # Don't conflict with standard logging
>>> from google.cloud.logging.handlers import CloudLoggingHandler, setup_logging
>>> client = google.cloud.logging.Client()
>>> handler = CloudLoggingHandler(client)
>>> logging.getLogger().setLevel(logging.INFO) # defaults to WARN
>>> setup_logging(handler)
>>> logging.error('bad news')
```

You can also exclude certain loggers:

```
>>> setup_logging(handler, excluded_loggers=('werkzeug',))
```

Python logging handler transports

The Python logging handler can use different transports. The default is `google.cloud.logging.handlers.BackgroundThreadTransport`.

1. `google.cloud.logging.handlers.BackgroundThreadTransport` this is the default. It writes entries on a background `python.threading.Thread`.
1. `google.cloud.logging.handlers.SyncTransport` this handler does a direct API call on each logging statement to write the entry.

Stackdriver Logging Client

Client for interacting with the Google Stackdriver Logging API.

class `google.cloud.logging.client.Client` (*project=None, credentials=None, http=None*)
 Bases: `google.cloud.client.JSONClient`

Client to bundle configuration needed for API requests.

Parameters

- **project** (*str*) – 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.

list_entries (*projects=None, filter_=None, order_by=None, page_size=None, page_token=None*)
 Return a page of log entries.

See: https://cloud.google.com/logging/docs/api/ref_v2beta1/rest/v2beta1/entries/list

Parameters

- **projects** (*list of strings*) – project IDs to include. If not passed, defaults to the project bound to the client.
- **filter** (*str*) – a filter expression. See: https://cloud.google.com/logging/docs/view/advanced_filters
- **order_by** (*str*) – One of `ASCENDING` or `DESCENDING`.
- **page_size** (*int*) – maximum number of entries to return, If not passed, defaults to a value set by the API.
- **page_token** (*str*) – opaque marker for the next “page” of entries. If not passed, the API will return the first page of entries.

Return type tuple, (list, str)

Returns list of `google.cloud.logging.entry.TextEntry`, plus a “next page token” string: if not `None`, indicates that more entries can be retrieved with another call (pass that value as `page_token`).

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

List metrics for the project associated with this client.

See: https://cloud.google.com/logging/docs/api/ref_v2beta1/rest/v2beta1/projects.metrics/list

Parameters

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

Return type tuple, (list, str)

Returns list of `google.cloud.logging.metric.Metric`, plus a “next page token” string: if not None, indicates that more metrics can be retrieved with another call (pass that value as `page_token`).

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

List sinks for the project associated with this client.

See: https://cloud.google.com/logging/docs/api/ref_v2beta1/rest/v2beta1/projects.sinks/list

Parameters

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

Return type tuple, (list, str)

Returns list of `google.cloud.logging.sink.Sink`, plus a “next page token” string: if not None, indicates that more sinks can be retrieved with another call (pass that value as `page_token`).

logger (*name*)

Creates a logger bound to the current client.

Parameters **name** (*str*) – the name of the logger to be constructed.

Return type `google.cloud.logging.logger.Logger`

Returns Logger created with the current client.

logging_api

Helper for logging-related API calls.

See: https://cloud.google.com/logging/docs/api/ref_v2beta1/rest/v2beta1/entries
https://cloud.google.com/logging/docs/api/ref_v2beta1/rest/v2beta1/projects.logs

metric (*name, filter_=None, description=''*)

Creates a metric bound to the current client.

Parameters

- **name** (*str*) – the name of the metric to be constructed.
- **filter** (*str*) – the advanced logs filter expression defining the entries tracked by the metric. If not passed, the instance should already exist, to be refreshed via `Metric.reload()`.
- **description** (*str*) – the description of the metric to be constructed. If not passed, the instance should already exist, to be refreshed via `Metric.reload()`.

Return type `google.cloud.logging.metric.Metric`

Returns Metric created with the current client.

metrics_api

Helper for log metric-related API calls.

See: https://cloud.google.com/logging/docs/api/ref_v2beta1/rest/v2beta1/projects.metrics

sink (*name*, *filter_=None*, *destination=None*)

Creates a sink bound to the current client.

Parameters

- **name** (*str*) – the name of the sink to be constructed.
- **filter** (*str*) – (optional) the advanced logs filter expression defining the entries exported by the sink. If not passed, the instance should already exist, to be refreshed via `Sink.reload()`.
- **destination** (*str*) – destination URI for the entries exported by the sink. If not passed, the instance should already exist, to be refreshed via `Sink.reload()`.

Return type `google.cloud.logging.sink.Sink`

Returns Sink created with the current client.

sinks_api

Helper for log sink-related API calls.

See: https://cloud.google.com/logging/docs/api/ref_v2beta1/rest/v2beta1/projects.sinks

57.1 Connection

Create / interact with Stackdriver Logging connections.

class `google.cloud.logging.connection.Connection` (*credentials=None*, *http=None*)

Bases: `google.cloud.connection.JSONConnection`

A connection to Google Stackdriver Logging 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://logging.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 = `'v2beta1'`

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

SCOPE = (`'https://www.googleapis.com/auth/logging.read'`, `'https://www.googleapis.com/auth/logging.write'`, `'https://www`

The scopes required for authenticating as a Logging consumer.

Logger

Define API Loggers.

class `google.cloud.logging.logger.Batch(logger, client)`
 Bases: `object`

Context manager: collect entries to log via a single API call.

Helper returned by `Logger.batch()`

Parameters

- **logger** (`google.cloud.logging.logger.Logger`) – the logger to which entries will be logged.
- **client** (`google.cloud.logging.client.Client`) – The client to use.

commit (`client=None`)

Send saved log entries as a single API call.

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

log_proto (`message, labels=None, insert_id=None, severity=None, http_request=None`)

Add a protobuf entry to be logged during `commit()`.

Parameters

- **message** (`protobuf message`) – the protobuf entry
- **labels** (`dict` or `NoneType`) – (optional) mapping of labels for the entry.
- **insert_id** (`string` or `NoneType`) – (optional) unique ID for log entry.
- **severity** (`string` or `NoneType`) – (optional) severity of event being logged.
- **http_request** (`dict` or `NoneType`) – (optional) info about HTTP request associated with the entry.

log_struct (`info, labels=None, insert_id=None, severity=None, http_request=None`)

Add a struct entry to be logged during `commit()`.

Parameters

- **info** (`dict`) – the struct entry
- **labels** (`dict` or `NoneType`) – (optional) mapping of labels for the entry.
- **insert_id** (`string` or `NoneType`) – (optional) unique ID for log entry.
- **severity** (`string` or `NoneType`) – (optional) severity of event being logged.

- **http_request** (dict or `NoneType`) – (optional) info about HTTP request associated with the entry.

log_text (*text*, *labels=None*, *insert_id=None*, *severity=None*, *http_request=None*)

Add a text entry to be logged during `commit()`.

Parameters

- **text** (*string*) – the text entry
- **labels** (dict or `NoneType`) – (optional) mapping of labels for the entry.
- **insert_id** (string or `NoneType`) – (optional) unique ID for log entry.
- **severity** (string or `NoneType`) – (optional) severity of event being logged.
- **http_request** (dict or `NoneType`) – (optional) info about HTTP request associated with the entry.

class `google.cloud.logging.logger.Logger` (*name*, *client*, *labels=None*)

Bases: `object`

Loggers represent named targets for log entries.

See: https://cloud.google.com/logging/docs/api/ref_v2beta1/rest/v2beta1/projects.logs

Parameters

- **name** (*string*) – the name of the logger
- **client** (`google.cloud.logging.client.Client`) – A client which holds credentials and project configuration for the logger (which requires a project).
- **labels** (dict or `NoneType`) – (optional) mapping of default labels for entries written via this logger.

batch (*client=None*)

Return a batch to use as a context manager.

Parameters **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.

client

Client bound to the logger.

delete (*client=None*)

API call: delete all entries in a logger via a DELETE request

See: https://cloud.google.com/logging/docs/api/ref_v2beta1/rest/v2beta1/projects.logs/delete

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

full_name

Fully-qualified name used in logging APIs

list_entries (*projects=None*, *filter_=None*, *order_by=None*, *page_size=None*, *page_token=None*)

Return a page of log entries.

See: https://cloud.google.com/logging/docs/api/ref_v2beta1/rest/v2beta1/entries/list

Parameters

- **projects** (*list of strings*) – project IDs to include. If not passed, defaults to the project bound to the client.
- **filter** (*string*) – a filter expression. See: https://cloud.google.com/logging/docs/view/advanced_filters
- **order_by** (*string*) – One of ASCENDING or DESCENDING.
- **page_size** (*int*) – maximum number of entries to return, If not passed, defaults to a value set by the API.
- **page_token** (*string*) – opaque marker for the next “page” of entries. If not passed, the API will return the first page of entries.

Return type tuple, (list, str)

Returns list of `google.cloud.logging.entry.TextEntry`, plus a “next page token” string: if not None, indicates that more entries can be retrieved with another call (pass that value as `page_token`).

log_proto (*message, client=None, labels=None, insert_id=None, severity=None, http_request=None*)

API call: log a protobuf message via a POST request

See: https://cloud.google.com/logging/docs/api/ref_v2beta1/rest/v2beta1/entries/write

Parameters

- **message** (*Protobuf message*) – the message to be logged
- **client** (*Client or NoneType*) – the client to use. If not passed, falls back to the `client` stored on the current logger.
- **labels** (*dict or NoneType*) – (optional) mapping of labels for the entry.
- **insert_id** (*string or NoneType*) – (optional) unique ID for log entry.
- **severity** (*string or NoneType*) – (optional) severity of event being logged.
- **http_request** (*dict or NoneType*) – (optional) info about HTTP request associated with the entry.

log_struct (*info, client=None, labels=None, insert_id=None, severity=None, http_request=None*)

API call: log a structured message via a POST request

See: https://cloud.google.com/logging/docs/api/ref_v2beta1/rest/v2beta1/entries/write

Parameters

- **info** (*dict*) – the log entry information
- **client** (*Client or NoneType*) – the client to use. If not passed, falls back to the `client` stored on the current logger.
- **labels** (*dict or NoneType*) – (optional) mapping of labels for the entry.
- **insert_id** (*string or NoneType*) – (optional) unique ID for log entry.
- **severity** (*string or NoneType*) – (optional) severity of event being logged.
- **http_request** (*dict or NoneType*) – (optional) info about HTTP request associated with the entry.

log_text (*text, client=None, labels=None, insert_id=None, severity=None, http_request=None*)

API call: log a text message via a POST request

See: https://cloud.google.com/logging/docs/api/ref_v2beta1/rest/v2beta1/entries/write

Parameters

- **text** (*text*) – the log message.
- **client** (*Client* or `NoneType`) – the client to use. If not passed, falls back to the `client` stored on the current logger.
- **labels** (dict or `NoneType`) – (optional) mapping of labels for the entry.
- **insert_id** (string or `NoneType`) – (optional) unique ID for log entry.
- **severity** (string or `NoneType`) – (optional) severity of event being logged.
- **http_request** (dict or `NoneType`) – (optional) info about HTTP request associated with the entry

path

URI path for use in logging APIs

project

Project bound to the logger.

Entries

Log entries within the Google Stackdriver Logging API.

```
class google.cloud.logging.entries.ProtobufEntry (payload, logger, insert_id=None,
                                                timestamp=None, labels=None, severity=None, http_request=None)
```

Bases: google.cloud.logging.entries._BaseEntry

Entry created with protoPayload.

See: https://cloud.google.com/logging/docs/api/ref_v2beta1/rest/v2beta1/LogEntry

```
parse_message (message)
```

Parse payload into a protobuf message.

Mutates the passed-in message in place.

Parameters *message* (Protobuf message) – the message to be logged

```
class google.cloud.logging.entries.StructEntry (payload, logger, insert_id=None, timestamp=None, labels=None, severity=None, http_request=None)
```

Bases: google.cloud.logging.entries._BaseEntry

Entry created with jsonPayload.

See: https://cloud.google.com/logging/docs/api/ref_v2beta1/rest/v2beta1/LogEntry

```
class google.cloud.logging.entries.TextEntry (payload, logger, insert_id=None, timestamp=None, labels=None, severity=None, http_request=None)
```

Bases: google.cloud.logging.entries._BaseEntry

Entry created with textPayload.

See: https://cloud.google.com/logging/docs/api/ref_v2beta1/rest/v2beta1/LogEntry

```
google.cloud.logging.entries.logger_name_from_path (path)
```

Validate a logger URI path and get the logger name.

Parameters *path* (str) – URI path for a logger API request.

Return type str

Returns Logger name parsed from path.

Raises ValueError if the path is ill-formed or if the project from the path does not agree with the project passed in.

Metrics

Define Stackdriver Logging API Metrics.

class `google.cloud.logging.metric.Metric` (*name*, *filter_=None*, *client=None*, *description=''*)
 Bases: `object`

Metrics represent named filters for log entries.

See: https://cloud.google.com/logging/docs/api/ref_v2beta1/rest/v2beta1/projects.metrics

Parameters

- **name** (*string*) – the name of the metric
- **filter** (*string*) – the advanced logs filter expression defining the entries tracked by the metric. If not passed, the instance should already exist, to be refreshed via `reload()`.
- **client** (`google.cloud.logging.client.Client`) – A client which holds credentials and project configuration for the metric (which requires a project).
- **description** (*string*) – an optional description of the metric.

client

Client bound to the logger.

create

 (*client=None*)

API call: create the metric via a PUT request

See: https://cloud.google.com/logging/docs/api/ref_v2beta1/rest/v2beta1/projects.metrics/create

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

delete

 (*client=None*)

API call: delete a metric via a DELETE request

See https://cloud.google.com/logging/docs/api/ref_v2beta1/rest/v2beta1/projects.metrics/delete

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

exists

 (*client=None*)

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

See https://cloud.google.com/logging/docs/api/ref_v2beta1/rest/v2beta1/projects.metrics/get

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

Return type `bool`

Returns Boolean indicating existence of the metric.

classmethod `from_api_repr` (*resource*, *client*)

Factory: construct a metric given its API representation

Parameters

- **resource** (*dict*) – metric resource representation returned from the API
- **client** (*google.cloud.logging.client.Client*) – Client which holds credentials and project configuration for the metric.

Return type *google.cloud.logging.metric.Metric*

Returns Metric parsed from *resource*.

full_name

Fully-qualified name used in metric APIs

path

URL path for the metric's APIs

project

Project bound to the logger.

reload (*client=None*)

API call: sync local metric configuration via a GET request

See https://cloud.google.com/logging/docs/api/ref_v2beta1/rest/v2beta1/projects.metrics/get

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

update (*client=None*)

API call: update metric configuration via a PUT request

See https://cloud.google.com/logging/docs/api/ref_v2beta1/rest/v2beta1/projects.metrics/update

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

Sinks

Define Stackdriver Logging API Sinks.

class `google.cloud.logging.sink.Sink` (*name*, *filter_=None*, *destination=None*, *client=None*)
 Bases: `object`

Sinks represent filtered exports for log entries.

See: https://cloud.google.com/logging/docs/api/ref_v2beta1/rest/v2beta1/projects.sinks

Parameters

- **name** (*string*) – the name of the sink
- **filter** (*string*) – the advanced logs filter expression defining the entries exported by the sink. If not passed, the instance should already exist, to be refreshed via `reload()`.
- **destination** (*string*) – destination URI for the entries exported by the sink. If not passed, the instance should already exist, to be refreshed via `reload()`.
- **client** (`google.cloud.logging.client.Client`) – A client which holds credentials and project configuration for the sink (which requires a project).

client

Client bound to the sink.

create

 (*client=None*)

API call: create the sink via a PUT request

See: https://cloud.google.com/logging/docs/api/ref_v2beta1/rest/v2beta1/projects.sinks/create

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

delete

 (*client=None*)

API call: delete a sink via a DELETE request

See https://cloud.google.com/logging/docs/api/ref_v2beta1/rest/v2beta1/projects.sinks/delete

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

exists

 (*client=None*)

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

See https://cloud.google.com/logging/docs/api/ref_v2beta1/rest/v2beta1/projects.sinks/get

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

Return type `bool`

Returns Boolean indicating existence of the sink.

classmethod `from_api_repr` (*resource*, *client*)

Factory: construct a sink given its API representation

Parameters

- **resource** (*dict*) – sink resource representation returned from the API
- **client** (*google.cloud.logging.client.Client*) – Client which holds credentials and project configuration for the sink.

Return type *google.cloud.logging.sink.Sink*

Returns Sink 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 sink APIs

path

URL path for the sink's APIs

project

Project bound to the sink.

reload (*client=None*)

API call: sync local sink configuration via a GET request

See https://cloud.google.com/logging/docs/api/ref_v2beta1/rest/v2beta1/projects.sinks/get

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

update (*client=None*)

API call: update sink configuration via a PUT request

See https://cloud.google.com/logging/docs/api/ref_v2beta1/rest/v2beta1/projects.sinks/update

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

Python Logging Module Handler

Python `logging` handlers for Google Cloud Logging.

```
class google.cloud.logging.handlers.handlers.CloudLoggingHandler (client,
                                                                name='python',
                                                                trans-
                                                                port=<class
                                                                'google.cloud.logging.handlers.transport
```

Bases: `logging.StreamHandler`

Python standard logging handler.

This handler can be used to route Python standard logging messages directly to the Stackdriver Logging API.

Note that this handler currently only supports a synchronous API call, which means each logging statement that uses this handler will require an API call.

Parameters

- **client** (*google.cloud.logging.client*) – the authenticated Google Cloud Logging client for this handler to use
- **name** (*str*) – the name of the custom log in Stackdriver Logging. Defaults to 'python'. The name of the Python logger will be represented in the `python_logger` field.
- **transport** (*type*) – Class for creating new transport objects. It should extend from the base *Transport* type and implement `:meth'.Transport.send'`. Defaults to *BackgroundThreadTransport*. The other option is *SyncTransport*.

Example:

```
import google.cloud.logging
from google.cloud.logging.handlers import CloudLoggingHandler

client = google.cloud.logging.Client()
handler = CloudLoggingHandler(client)

cloud_logger = logging.getLogger('cloudLogger')
cloud_logger.setLevel(logging.INFO)
cloud_logger.addHandler(handler)

cloud_logger.error('bad news') # API call
```

emit (*record*)

Actually log the specified logging record.

Overrides the default emit behavior of `StreamHandler`.

See: <https://docs.python.org/2/library/logging.html#handler-objects>

Parameters **record** (`logging.LogRecord`) – The record to be logged.

`google.cloud.logging.handlers.handlers.setup_logging` (*handler*, *excluded_loggers*=('google.cloud', 'oauth2client'))

Attach the CloudLogging handler to the Python root logger

Excludes loggers that this library itself uses to avoid infinite recursion.

Parameters

- **handler** (`logging.handler`) – the handler to attach to the global handler
- **excluded_loggers** (*tuple*) – The loggers to not attach the handler to. This will always include the loggers in the path of the logging client itself.

Example:

```
import logging
import google.cloud.logging
from google.cloud.logging.handlers import CloudLoggingHandler

client = google.cloud.logging.Client()
handler = CloudLoggingHandler(client)
google.cloud.logging.setup_logging(handler)
logging.getLogger().setLevel(logging.DEBUG)

logging.error('bad news') # API call
```

Python Logging Handler Sync Transport

Transport for Python logging handler.

Logs directly to the the Stackdriver Logging API with a synchronous call.

class `google.cloud.logging.handlers.transports.sync.SyncTransport` (*client, name*)

Bases: `google.cloud.logging.handlers.transports.base.Transport`

Basic synchronous transport.

Uses this library's Logging client to directly make the API call.

send (*record, message*)

Overrides `transport.send()`.

Parameters

- **record** (`logging.LogRecord`) – Python log record that the handler was called with.
- **message** (`str`) – The message from the `LogRecord` after being formatted by the associated log formatters.

Python Logging Handler Threaded Transport

Transport for Python logging handler

Uses a background worker to log to Stackdriver Logging asynchronously.

class `google.cloud.logging.handlers.transports.background_thread.BackgroundThreadTransport` (*class name*)

Bases: `google.cloud.logging.handlers.transports.base.Transport`

Asynchronous transport that uses a background thread.

Writes logging entries as a batch process.

send (*record, message*)

Overrides `Transport.send()`.

Parameters

- **record** (`logging.LogRecord`) – Python log record that the handler was called with.
- **message** (*str*) – The message from the `LogRecord` after being formatted by the associated log formatters.

Python Logging Handler Sync Transport

Module containing base class for logging transport.

class `google.cloud.logging.handlers.transports.base.Transport`

Bases: `object`

Base class for Google Cloud Logging handler transports.

Subclasses of `Transport` must have constructors that accept a client and name object, and must override `send()`.

send (`record`, `message`)

Transport send to be implemented by subclasses.

Parameters

- **record** (`logging.LogRecord`) – Python log record that the handler was called with.
- **message** (`str`) – The message from the `LogRecord` after being formatted by the associated log formatters.

Using the API

66.1 Authentication and Configuration

- For an overview of authentication in `google-cloud-python`, see [Authentication](#).
- In addition to any authentication configuration, you should also set the `GOOGLE_CLOUD_PROJECT` environment variable for the project you'd like to interact with. If you are Google App Engine or Google Compute Engine this will be detected automatically.
- After configuring your environment, create a `Client`

```
>>> from google.cloud import error_reporting
>>> client = error_reporting.Client()
```

or pass in credentials and project explicitly

```
>>> from google.cloud import error_reporting
>>> client = error_reporting.Client(project='my-project', credentials=creds)
```

Error Reporting associates errors with a service, which is an identifier for an executable, App Engine service, or job. The default service is “python”, but a default can be specified for the client on construction time. You can also optionally specify a version for that service, which defaults to “default.”

```
>>> from google.cloud import error_reporting
>>> client = error_reporting.Client(project='my-project',
...                               service="login_service",
...                               version="0.1.0")
```

66.2 Reporting an exception

Report a stacktrace to Stackdriver Error Reporting after an exception

```
>>> from google.cloud import error_reporting
>>> client = error_reporting.Client()
>>> try:
>>>     raise NameError
>>> except Exception:
>>>     client.report_exception()
```

By default, the client will report the error using the service specified in the client's constructor, or the default service of “python”.

The user and HTTP context can also be included in the exception. The HTTP context can be constructed using `google.cloud.error_reporting.HTTPContext`. This will be used by Stackdriver Error Reporting to help group exceptions.

```
>>> from google.cloud import error_reporting
>>> client = error_reporting.Client()
>>> user = 'example@gmail.com'
>>> http_context = HTTPContext(method='GET', url='/', userAgent='test agent',
...                             referrer='example.com', responseStatusCode=500,
...                             remote_ip='1.2.3.4')
>>> try:
>>>     raise NameError
>>> except Exception:
>>>     client.report_exception(http_context=http_context, user=user)
```

66.3 Reporting an error without an exception

Errors can also be reported to Stackdriver Error Reporting outside the context of an exception. The library will include the file path, function name, and line number of the location where the error was reported.

```
>>> from google.cloud import error_reporting
>>> client = error_reporting.Client()
>>> error_reporting.report("Found an error!")
```

Similarly to reporting an exception, the user and HTTP context can be provided:

```
>>> from google.cloud import error_reporting
>>> client = error_reporting.Client()
>>> user = 'example@gmail.com'
>>> http_context = HTTPContext(method='GET', url='/', userAgent='test agent',
...                             referrer='example.com', responseStatusCode=500,
...                             remote_ip='1.2.3.4')
>>> error_reporting.report("Found an error!", http_context=http_context, user=user)
```

Error Reporting Client

Client for interacting with the Stackdriver Logging API

```
class google.cloud.error_reporting.client.Client (project=None, credentials=None,
                                                http=None, service=None, version=None)
```

Bases: `object`

Error Reporting client. Currently Error Reporting is done by creating a Logging client.

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.
- **service** (*str*) – An identifier of the service, such as the name of the executable, job, or Google App Engine service name. This field is expected to have a low number of values that are relatively stable over time, as opposed to version, which can be changed whenever new code is deployed.
- **version** (*str*) – Represents the source code version that the developer provided, which could represent a version label or a Git SHA-1 hash, for example. If the developer did not provide a version, the value is set to default.

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

report (*message*, *http_context=None*, *user=None*)

Reports a message to Stackdriver Error Reporting <https://cloud.google.com/error-reporting/docs/formatting-error-messages>

type message `str`

param message A user-supplied message to report

type http_context `:class'google.cloud.error_reporting.HTTPContext'`

param http_context The HTTP request which was processed when the error was triggered.

type user `string`

param user

The user who caused or was affected by the crash. This can be a user ID, an email address, or an arbitrary token that uniquely identifies the user. When sending an error report, leave this field empty if the user was not logged in. In this case the Error Reporting system will use other data, such as remote IP address, to distinguish affected users.

Example::

```
>>> client.report("Something went wrong!")
```

report_exception (*http_context=None, user=None*)

Reports the details of the latest exceptions to Stackdriver Error Reporting.

Parameters

- **http_context** (*:class 'google.cloud.error_reporting.HTTPContext'*) – The HTTP request which was processed when the error was triggered.
- **user** (*string*) –

The user who caused or was affected by the crash. This can be a user ID, an email address, or an arbitrary token that uniquely identifies the user. When sending an error report, leave this field empty if the user was not logged in. In this case the Error Reporting system will use other data, such as remote IP address, to distinguish affected users.

Example:

```
>>> try:
>>>     raise NameError
>>> except Exception:
>>>     client.report_exception()
```

```
class google.cloud.error_reporting.client.HTTPContext (method=None, url=None,
                                                       user_agent=None, re-
                                                       ferrer=None, re-
                                                       sponse_status_code=None,
                                                       remote_ip=None)
```

Bases: `object`

HTTPContext defines an object that captures the parameter for the `HttpRequest` part of Error Reporting API

Parameters

- **method** (*string*) – The type of HTTP request, such as GET, POST, etc.
- **url** (*string*) – The URL of the request
- **user_agent** (*string*) – The user agent information that is provided with the request.
- **referrer** (*string*) – The referrer information that is provided with the request.
- **response_status_code** (*int*) – The HTTP response status code for the request.
- **remote_ip** (*string*) – The IP address from which the request originated. This can be IPv4, IPv6, or a token which is derived from the IP address, depending on the data that has been provided in the error report.

Using the API

68.1 Introduction

With the Stackdriver Monitoring API, you can work with Stackdriver metric data pertaining to monitored resources in Google Cloud Platform (GCP) or elsewhere.

Essential concepts:

- Metric data is associated with a **monitored resource**. A monitored resource has a *resource type* and a set of *resource labels* — key-value pairs — that identify the particular resource.
- A **metric** further identifies the particular kind of data that is being collected. It has a *metric type* and a set of *metric labels* that, when combined with the resource labels, identify a particular time series.
- A **time series** is a collection of data points associated with points or intervals in time.

Please refer to the documentation for the [Stackdriver Monitoring API](#) for more information.

At present, this client library supports the following features of the API:

- Querying of time series.
- Querying of metric descriptors and monitored resource descriptors.
- Creation and deletion of metric descriptors for custom metrics.
- (Writing of custom metric data will be coming soon.)

68.2 The Stackdriver Monitoring Client Object

The Stackdriver Monitoring client library generally makes its functionality available as methods of the monitoring *Client* class. A *Client* instance holds authentication credentials and the ID of the target project with which the metric data of interest is associated. This project ID will often refer to a [Stackdriver account](#) binding multiple GCP projects and AWS accounts. It can also simply be the ID of a monitored project.

Most often the authentication credentials will be determined implicitly from your environment. See [Authentication](#) for more information.

It is thus typical to create a client object as follows:

```
>>> from google.cloud import monitoring
>>> client = monitoring.Client(project='target-project')
```

If you are running in Google Compute Engine or Google App Engine, the current project is the default target project. This default can be further overridden with the `GOOGLE_CLOUD_PROJECT` environment variable. Using the default target project is even easier:

```
>>> client = monitoring.Client()
```

If necessary, you can pass in `credentials` and `project` explicitly:

```
>>> client = monitoring.Client(project='target-project', credentials=...)
```

68.3 Monitored Resource Descriptors

The available monitored resource types are defined by *monitored resource descriptors*. You can fetch a list of these with the `list_resource_descriptors()` method:

```
>>> for descriptor in client.list_resource_descriptors():
...     print(descriptor.type)
```

Each *ResourceDescriptor* has a type, a display name, a description, and a list of *LabelDescriptor* instances. See the documentation about [Monitored Resources](#) for more information.

68.4 Metric Descriptors

The available metric types are defined by *metric descriptors*. They include [platform metrics](#), [agent metrics](#), and [custom metrics](#). You can list all of these with the `list_metric_descriptors()` method:

```
>>> for descriptor in client.list_metric_descriptors():
...     print(descriptor.type)
```

See *MetricDescriptor* and the [Metric Descriptors API](#) documentation for more information.

You can create new metric descriptors to define custom metrics in the `custom.googleapis.com` namespace. You do this by creating a *MetricDescriptor* object using the client's `metric_descriptor()` factory and then calling the object's `create()` method:

```
>>> from google.cloud.monitoring import MetricKind, ValueType
>>> descriptor = client.metric_descriptor(
...     'custom.googleapis.com/my_metric',
...     metric_kind=MetricKind.GAUGE,
...     value_type=ValueType.DOUBLE,
...     description='This is a simple example of a custom metric.')
>>> descriptor.create()
```

You can delete such a metric descriptor as follows:

```
>>> descriptor = client.metric_descriptor(
...     'custom.googleapis.com/my_metric')
>>> descriptor.delete()
```

To define a custom metric parameterized by one or more labels, you must build the appropriate *LabelDescriptor* objects and include them in the *MetricDescriptor* object before you call *create()*:

```
>>> from google.cloud.monitoring import LabelDescriptor, LabelValueType
>>> label = LabelDescriptor('response_code', LabelValueType.INT64,
...                         description='HTTP status code')
...
>>> descriptor = client.metric_descriptor(
...     'custom.googleapis.com/my_app/response_count',
...     metric_kind=MetricKind.CUMULATIVE,
...     value_type=ValueTypes.INT64,
...     labels=[label],
...     description='Cumulative count of HTTP responses.')
>>> descriptor.create()
```

68.5 Groups

A group is a dynamic collection of *monitored resources* whose membership is defined by a *filter*. These groups are usually created via the *Stackdriver* dashboard. You can list all the groups in a project with the *list_groups()* method:

```
>>> for group in client.list_groups():
...     print(group.id, group.display_name, group.parent_id)
('a001', 'Production', None)
('a002', 'Front-end', 'a001')
('1003', 'Back-end', 'a001')
```

See *Group* and the API documentation for *Groups* and *Group members* for more information.

You can get a specific group based on its ID as follows:

```
>>> group = client.fetch_group('a001')
```

You can get the current members of this group using the *list_members()* method:

```
>>> for member in group.list_members():
...     print(member)
```

Passing in *end_time* and *start_time* to the above method will return historical members based on the current filter of the group. The group membership changes over time, as *monitored resources* come and go, and as they change properties.

You can create new groups to define new collections of *monitored resources*. You do this by creating a *Group* object using the client's *group()* factory and then calling the object's *create()* method:

```
>>> filter_string = 'resource.zone = "us-central1-a"'
>>> group = client.group(
...     display_name='My group',
...     filter_string=filter_string,
...     parent_id='a001',
...     is_cluster=True)
>>> group.create()
>>> group.id
'1234'
```

You can further manipulate an existing group by first initializing a `Group` object with its ID or name, and then calling various methods on it.

Delete a group:

```
>>> group = client.group('1234')
>>> group.exists()
True
>>> group.delete()
```

Update a group:

```
>>> group = client.group('1234')
>>> group.exists()
True
>>> group.reload()
>>> group.display_name = 'New Display Name'
>>> group.update()
```

68.6 Time Series Queries

A time series includes a collection of data points and a set of resource and metric label values. See [TimeSeries](#) and the [Time Series API](#) documentation for more information.

While you can obtain time series objects by iterating over a `Query` object, usually it is more useful to retrieve time series data in the form of a `pandas.DataFrame`, where each column corresponds to a single time series. For this, you must have `pandas` installed; it is not a required dependency of `google-cloud-python`.

You can display CPU utilization across your GCE instances during the last five minutes as follows:

```
>>> METRIC = 'compute.googleapis.com/instance/cpu/utilization'
>>> query = client.query(METRIC, minutes=5)
>>> print(query.as_dataframe())
```

`Query` objects provide a variety of methods for refining the query. You can request temporal alignment and cross-series reduction, and you can filter by label values. See the client `query()` method and the `Query` class for more information.

For example, you can display CPU utilization during the last hour across GCE instances with names beginning with "mycluster-", averaged over five-minute intervals and aggregated per zone, as follows:

```
>>> from google.cloud.monitoring import Aligner, Reducer
>>> METRIC = 'compute.googleapis.com/instance/cpu/utilization'
>>> query = (client.query(METRIC, hours=1)
...         .select_metrics(instance_name_prefix='mycluster-')
...         .align(Aligner.ALIGN_MEAN, minutes=5)
...         .reduce(Reducer.REDUCE_MEAN, 'resource.zone'))
>>> print(query.as_dataframe())
```

Stackdriver Monitoring Client

Client for interacting with the Google Stackdriver Monitoring API (V3).

Example:

```
>>> from google.cloud import monitoring
>>> client = monitoring.Client()
>>> query = client.query(minutes=5)
>>> print(query.as_dataframe()) # Requires pandas.
```

At present, the client supports querying of time series, metric descriptors, and monitored resource descriptors.

class `google.cloud.monitoring.client.Client` (*project=None, credentials=None, http=None*)

Bases: `google.cloud.client.JSONClient`

Client to bundle configuration needed for API requests.

Parameters

- **project** (*string*) – The target project. 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.

fetch_group (*group_id*)

Fetch a group from the API based on it's ID.

Example:

```
>>> try:
>>>     group = client.fetch_group('1234')
>>> except google.cloud.exceptions.NotFound:
>>>     print('That group does not exist!')
```

Parameters `group_id` (*string*) – The ID of the group.

Return type `Group`

Returns The group instance.

Raises `google.cloud.exceptions.NotFound` if the group is not found.

fetch_metric_descriptor (*metric_type*)

Look up a metric descriptor by type.

Example:

```
>>> METRIC = 'compute.googleapis.com/instance/cpu/utilization'
>>> print(client.fetch_metric_descriptor(METRIC))
```

Parameters `metric_type` (*string*) – The metric type name.

Return type `MetricDescriptor`

Returns The metric descriptor instance.

Raises `google.cloud.exceptions.NotFound` if the metric descriptor is not found.

fetch_resource_descriptor (*resource_type*)

Look up a monitored resource descriptor by type.

Example:

```
>>> print(client.fetch_resource_descriptor('gce_instance'))
```

Parameters `resource_type` (*string*) – The resource type name.

Return type `ResourceDescriptor`

Returns The resource descriptor instance.

Raises `google.cloud.exceptions.NotFound` if the resource descriptor is not found.

group (*group_id=None, display_name=None, parent_id=None, filter_string=None, is_cluster=False*)

Factory constructor for group object.

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

Parameters

- **group_id** (*string or None*) – The ID of the group.
- **display_name** (*string or None*) – A user-assigned name for this group, used only for display purposes.
- **parent_id** (*string or None*) – The ID of the group's parent, if it has one.
- **filter_string** (*string or None*) – The filter string used to determine which monitored resources belong to this group.
- **is_cluster** (*boolean*) – If true, the members of this group are considered to be a cluster. The system can perform additional analysis on groups that are clusters.

Return type `Group`

Returns The group created with the passed-in arguments.

Raises `ValueError` if both `group_id` and `name` are specified.

list_groups()

List all groups for the project.

Example:

```
>>> for group in client.list_groups():
...     print((group.display_name, group.name))
```

Return type list of *Group*

Returns A list of group instances.

list_metric_descriptors (*filter_string=None, type_prefix=None*)

List all metric descriptors for the project.

Examples:

```
>>> for descriptor in client.list_metric_descriptors():
...     print(descriptor.type)

>>> for descriptor in client.list_metric_descriptors(
...     type_prefix='custom.'):
...     print(descriptor.type)
```

Parameters

- **filter_string** (*string or None*) – An optional filter expression describing the metric descriptors to be returned. See the [filter documentation](#).
- **type_prefix** (*string or None*) – An optional prefix constraining the selected metric types. This adds `metric.type = starts_with("<prefix>")` to the filter.

Return type list of *MetricDescriptor*

Returns A list of metric descriptor instances.

list_resource_descriptors (*filter_string=None*)

List all monitored resource descriptors for the project.

Example:

```
>>> for descriptor in client.list_resource_descriptors():
...     print(descriptor.type)
```

Parameters **filter_string** (*string or None*) – An optional filter expression describing the resource descriptors to be returned. See the [filter documentation](#).

Return type list of *ResourceDescriptor*

Returns A list of resource descriptor instances.

static metric (*type_, labels*)

Factory for constructing metric objects.

Metric objects are typically created to write custom metric values. The type should match the metric type specified in the *MetricDescriptor* used to create the custom metric:

```
>>> metric = client.metric('custom.googleapis.com/my_metric',
...                         labels={
...                             'status': 'successful',
...                         })
```

Parameters

- **type** (*string*) – The metric type name.
- **labels** (*dict*) – A mapping from label names to values for all labels enumerated in the associated *MetricDescriptor*.

Return type *Metric*

Returns The metric object.

```
metric_descriptor (type_, metric_kind='METRIC_KIND_UNSPECIFIED',
                    value_type='VALUE_TYPE_UNSPECIFIED', labels=(), unit='', de-
                    scription='', display_name='')
```

Construct a metric descriptor object.

Metric descriptors specify the schema for a particular metric type.

This factory method is used most often in conjunction with the metric descriptor *create()* method to define custom metrics:

```
>>> descriptor = client.metric_descriptor(
...     'custom.googleapis.com/my_metric',
...     metric_kind=MetricKind.GAUGE,
...     value_type=ValueTypes.DOUBLE,
...     description='This is a simple example of a custom metric.')
>>> descriptor.create()
```

Here is an example where the custom metric is parameterized by a metric label:

```
>>> label = LabelDescriptor('response_code', LabelValueType.INT64,
...                         description='HTTP status code')
>>> descriptor = client.metric_descriptor(
...     'custom.googleapis.com/my_app/response_count',
...     metric_kind=MetricKind.CUMULATIVE,
...     value_type=ValueTypes.INT64,
...     labels=[label],
...     description='Cumulative count of HTTP responses.')
>>> descriptor.create()
```

Parameters

- **type** (*string*) – The metric type including a DNS name prefix. For example: "custom.googleapis.com/my_metric"
- **metric_kind** (*string*) – The kind of measurement. It must be one of *MetricKind.GAUGE*, *MetricKind.DELTA*, or *MetricKind.CUMULATIVE*. See *MetricKind*.
- **value_type** (*string*) – The value type of the metric. It must be one of *ValueType.BOOL*, *ValueType.INT64*, *ValueType.DOUBLE*, *ValueType.STRING*, or *ValueType.DISTRIBUTION*. See *ValueType*.
- **labels** (list of *LabelDescriptor*) – A sequence of zero or more label descriptors specifying the labels used to identify a specific instance of this metric.

- **unit** (*string*) – An optional unit in which the metric value is reported.
- **description** (*string*) – An optional detailed description of the metric.
- **display_name** (*string*) – An optional concise name for the metric.

Return type `MetricDescriptor`

Returns The metric descriptor created with the passed-in arguments.

query (*metric_type*='compute.googleapis.com/instance/cpu/utilization', *end_time*=None, *days*=0, *hours*=0, *minutes*=0)

Construct a query object for retrieving metric data.

Example:

```
>>> query = client.query(minutes=5)
>>> print(query.as_dataframe()) # Requires pandas.
```

Parameters

- **metric_type** (*string*) – The metric type name. The default value is `Query.DEFAULT_METRIC_TYPE`, but please note that this default value is provided only for demonstration purposes and is subject to change. See the [supported metrics](#).
- **end_time** (`datetime.datetime` or None) – The end time (inclusive) of the time interval for which results should be returned, as a datetime object. The default is the start of the current minute.

The start time (exclusive) is determined by combining the values of `days`, `hours`, and `minutes`, and subtracting the resulting duration from the end time.

It is also allowed to omit the end time and duration here, in which case `select_interval()` must be called before the query is executed.

- **days** (*integer*) – The number of days in the time interval.
- **hours** (*integer*) – The number of hours in the time interval.
- **minutes** (*integer*) – The number of minutes in the time interval.

Return type `Query`

Returns The query object.

Raises `ValueError` if `end_time` is specified but `days`, `hours`, and `minutes` are all zero. If you really want to specify a point in time, use `select_interval()`.

static resource (*type_*, *labels*)

Factory for constructing monitored resource objects.

A monitored resource object (`Resource`) is typically used to create a `TimeSeries` object.

For a list of possible monitored resource types and their associated labels, see:

<https://cloud.google.com/monitoring/api/resources>

Parameters

- **type** (*string*) – The monitored resource type name.
- **labels** (*dict*) – A mapping from label names to values for all labels enumerated in the associated `ResourceDescriptor`, except that `project_id` can and should be omitted when writing time series data.

Return type *Resource*

Returns A monitored resource object.

static time_series (*metric, resource, value, end_time=None, start_time=None*)
Construct a time series object for a single data point.

Note: While *TimeSeries* objects returned by the API typically have multiple data points, *TimeSeries* objects sent to the API must have at most one point.

For example:

```
>>> timeseries = client.time_series(metric, resource, 1.23,
...                               end_time=end)
```

For more information, see:

https://cloud.google.com/monitoring/api/ref_v3/rest/v3/TimeSeries

Parameters

- **metric** (*Metric*) – A *Metric*.
- **resource** (*Resource*) – A *Resource* object.
- **value** (*bool, int, string, or float*) – The value of the data point to create for the *TimeSeries*.

Note: The Python type of the value will determine the `ValueType` sent to the API, which must match the value type specified in the metric descriptor. For example, a Python float will be sent to the API as a `ValueType.DOUBLE`.

- **end_time** (*datetime*) – The end time for the point to be included in the time series. Assumed to be UTC if no time zone information is present. Defaults to the current time, as obtained by calling `datetime.datetime.utcnow()`.
- **start_time** (*datetime*) – The start time for the point to be included in the time series. Assumed to be UTC if no time zone information is present Defaults to `None`. If the start time is unspecified, the API interprets the start time to be the same as the end time.

Return type *TimeSeries*

Returns A time series object.

69.1 Connection

Create / interact with Stackdriver Monitoring connections.

class `google.cloud.monitoring.connection.Connection` (*credentials=None, http=None*)
Bases: `google.cloud.connection.JSONConnection`

A connection to Google Stackdriver Monitoring 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://monitoring.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 = `'v3'`

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

SCOPE = (`'https://www.googleapis.com/auth/monitoring.read'`, `'https://www.googleapis.com/auth/monitoring'`, `'https://www.googleapis.com/auth/monitoring.read'`)

The scopes required for authenticating as a Monitoring consumer.

Metric Descriptors

Metric Descriptors for the Google Stackdriver Monitoring API (V3).

class `google.cloud.monitoring.metric.Metric`
 Bases: `google.cloud.monitoring.metric.Metric`

A specific metric identified by specifying values for all labels.

The preferred way to construct a metric object is using the `metric()` factory method of the `Client` class.

Parameters

- **type** (*string*) – The metric type name.
- **labels** (*dict*) – A mapping from label names to values for all labels enumerated in the associated `MetricDescriptor`.

class `google.cloud.monitoring.metric.MetricDescriptor` (*client*, *type_*, *metric_kind*=`'METRIC_KIND_UNSPECIFIED'`, *value_type*=`'VALUE_TYPE_UNSPECIFIED'`, *labels*=(), *unit*=',', *description*=',', *display_name*=',', *name*=None)

Bases: `object`

Specification of a metric type and its schema.

The preferred way to construct a metric descriptor object is using the `metric_descriptor()` factory method of the `Client` class.

Parameters

- **client** (`google.cloud.monitoring.client.Client`) – A client for operating on the metric descriptor.
- **type** (*string*) – The metric type including a DNS name prefix. For example: `"compute.googleapis.com/instance/cpu/utilization"`
- **metric_kind** (*string*) – The kind of measurement. It must be one of `MetricKind.GAUGE`, `MetricKind.DELTA`, or `MetricKind.CUMULATIVE`. See `MetricKind`.
- **value_type** (*string*) – The value type of the metric. It must be one of `ValueType.BOOL`, `ValueType.INT64`, `ValueType.DOUBLE`, `ValueType.STRING`, or `ValueType.DISTRIBUTION`. See `ValueType`.
- **labels** (list of `LabelDescriptor`) – A sequence of zero or more label descriptors specifying the labels used to identify a specific instance of this metric.

- **unit** (*string*) – An optional unit in which the metric value is reported.
- **description** (*string*) – An optional detailed description of the metric.
- **display_name** (*string*) – An optional concise name for the metric.
- **name** (*string or None*) – The “resource name” of the metric descriptor. For example: "projects/<project_id>/metricDescriptors/<type>". As retrieved from the service, this will always be specified. You can and should omit it when constructing an instance for the purpose of creating a new metric descriptor.

create()

Create a new metric descriptor based on this object.

Example:

```
>>> descriptor = client.metric_descriptor(  
...     'custom.googleapis.com/my_metric',  
...     metric_kind=MetricKind.GAUGE,  
...     value_type=ValueTypes.DOUBLE,  
...     description='This is a simple example of a custom metric.')  
>>> descriptor.create()
```

The metric kind must not be `MetricKind.METRIC_KIND_UNSPECIFIED`, and the value type must not be `ValueType.VALUE_TYPE_UNSPECIFIED`.

The name attribute is ignored in preparing the creation request. All attributes are overwritten by the values received in the response (normally affecting only name).

delete()

Delete the metric descriptor identified by this object.

Example:

```
>>> descriptor = client.metric_descriptor(  
...     'custom.googleapis.com/my_metric')  
>>> descriptor.delete()
```

Only the `client` and `type` attributes are used.

class google.cloud.monitoring.metric.**MetricKind**

Bases: `object`

Choices for the kind of measurement.

METRIC_KIND_UNSPECIFIED = 'METRIC_KIND_UNSPECIFIED'

Note: An unspecified kind is not allowed in metric descriptors.

class google.cloud.monitoring.metric.**ValueType**

Bases: `object`

Choices for the metric value type.

VALUE_TYPE_UNSPECIFIED = 'VALUE_TYPE_UNSPECIFIED'

Note: An unspecified type is not allowed in metric descriptors.

Monitored Resource Descriptors

Monitored Resource Descriptors for the Google Stackdriver Monitoring API (V3).

class `google.cloud.monitoring.resource.Resource`

Bases: `google.cloud.monitoring.resource.Resource`

A monitored resource identified by specifying values for all labels.

The preferred way to construct a resource object is using the `resource()` factory method of the `Client` class.

Parameters

- **type** (*string*) – The resource type name.
- **labels** (*dict*) – A mapping from label names to values for all labels enumerated in the associated `ResourceDescriptor`.

class `google.cloud.monitoring.resource.ResourceDescriptor` (*name*, *type_*, *display_name*, *description*, *labels*)

Bases: `object`

Specification of a monitored resource type and its schema.

Parameters

- **name** (*string*) – The “resource name” of the monitored resource descriptor: `projects/<project_id>/monitoredResourceDescriptors/<type>`
- **type** (*string*) – The monitored resource type. For example: `"gce_instance"`
- **display_name** (*string*) – A concise name that might be displayed in user interfaces.
- **description** (*string*) – A detailed description that might be used in documentation.
- **labels** (list of `LabelDescriptor`) – A sequence of label descriptors specifying the labels used to identify a specific instance of this monitored resource.

Groups

Groups for the Google Stackdriver Monitoring API (V3).

```
class google.cloud.monitoring.group.Group(client, group_id=None, display_name=None,
                                           parent_id=None, filter_string=None,
                                           is_cluster=False)
```

Bases: `object`

A dynamic collection of monitored resources.

Parameters

- **client** (*google.cloud.monitoring.client.Client*) – A client for operating on the metric descriptor.
- **group_id** (*string or None*) – The ID of the group.
- **display_name** (*string or None*) – A user-assigned name for this group, used only for display purposes.
- **parent_id** (*string or None*) – The ID of the group's parent, if it has one.
- **filter_string** (*string or None*) – The filter string used to determine which monitored resources belong to this group.
- **is_cluster** (*boolean*) – If true, the members of this group are considered to be a cluster. The system can perform additional analysis on groups that are clusters.

`create()`

Create a new group based on this object via a POST request.

Example:

```
>>> filter_string = 'resource.type = "gce_instance"'
>>> group = client.group(
...     display_name='My group',
...     filter_string=filter_string,
...     parent_id='5678',
...     is_cluster=True)
>>> group.create()
```

The name attribute is ignored in preparing the creation request. All attributes are overwritten by the values received in the response (normally affecting only name).

`delete()`

Delete the group via a DELETE request.

Example:

```
>>> group = client.group('1234')
>>> group.delete()
```

Only the `client` and `name` attributes are used.

Warning: This method will fail for groups that have one or more children groups.

`exists()`

Test for the existence of the group via a GET request.

Return type `bool`

Returns Boolean indicating existence of the group.

`fetch_parent()`

Returns the parent group of this group via a GET request.

Return type `Group` or `None`

Returns The parent of the group.

`id`

Returns the group ID.

Return type `str` or `None`

Returns the ID of the group based on it's name.

`list_ancestors()`

Lists all ancestors of this group via a GET request.

The groups are returned in order, starting with the immediate parent and ending with the most distant ancestor. If the specified group has no immediate parent, the results are empty.

Return type list of `Group`

Returns A list of group instances.

`list_children()`

Lists all children of this group via a GET request.

Returns groups whose `parent_name` field contains the group name. If no groups have this parent, the results are empty.

Return type list of `Group`

Returns A list of group instances.

`list_descendants()`

Lists all descendants of this group via a GET request.

This returns a superset of the results returned by the `children()` method, and includes children-of-children, and so forth.

Return type list of `Group`

Returns A list of group instances.

`list_members(filter_string=None, end_time=None, start_time=None)`

Lists all members of this group via a GET request.

If no `end_time` is provided then the group membership over the last minute is returned.

Example:

```
>>> for member in group.list_members():
...     print member
```

List members that are Compute Engine VM instances:

```
>>> filter_string = 'resource.type = "gce_instance"'
>>> for member in group.list_members(filter_string=filter_string):
...     print member
```

List historical members that existed between 4 and 5 hours ago:

```
>>> import datetime
>>> t1 = datetime.datetime.utcnow() - datetime.timedelta(hours=4)
>>> t0 = t1 - datetime.timedelta(hours=1)
>>> for member in group.list_members(end_time=t1, start_time=t0):
...     print member
```

Parameters

- **filter_string** (*string* or *None*) – An optional list filter describing the members to be returned. The filter may reference the type, labels, and metadata of monitored resources that comprise the group. See the [filter documentation](#).
- **end_time** (*datetime.datetime* or *None*) – The end time (inclusive) of the time interval for which results should be returned, as a datetime object. If *start_time* is specified, then this must also be specified.
- **start_time** (*datetime.datetime* or *None*) – The start time (exclusive) of the time interval for which results should be returned, as a datetime object.

Return type list of *Resource*

Returns A list of resource instances.

Raises *ValueError* if the *start_time* is specified, but the *end_time* is missing.

name

Returns the fully qualified name of the group.

Return type str or *None*

Returns The fully qualified name of the group in the format “projects/<project>/groups/<id>”.

parent_name

Returns the fully qualified name of the parent group.

Return type str or *None*

Returns The fully qualified name of the parent group.

path

URL path to this group.

Return type str

Returns the path based on project and group name.

Raises *ValueError* if *name* is not specified.

reload()

Sync local group information via a GET request.

Warning: This will overwrite any local changes you've made and not saved via `update()`.

update()

Update the group via a PUT request.

Time Series Query

Time series query for the [Google Stackdriver Monitoring API \(V3\)](#).

class `google.cloud.monitoring.query.Aligner`

Bases: `object`

Allowed values for the supported aligners.

class `google.cloud.monitoring.query.Query` (*client, metric_type='compute.googleapis.com/instance/cpu/utilization', end_time=None, days=0, hours=0, minutes=0*)

Bases: `object`

Query object for retrieving metric data.

The preferred way to construct a query object is using the `query()` method of the `Client` class.

Parameters

- **client** (`google.cloud.monitoring.client.Client`) – The client to use.
- **metric_type** (`string`) – The metric type name. The default value is `Query.DEFAULT_METRIC_TYPE`, but please note that this default value is provided only for demonstration purposes and is subject to change. See the [supported metrics](#).
- **end_time** (`datetime.datetime` or `None`) – The end time (inclusive) of the time interval for which results should be returned, as a `datetime` object. The default is the start of the current minute.

The start time (exclusive) is determined by combining the values of `days`, `hours`, and `minutes`, and subtracting the resulting duration from the end time.

It is also allowed to omit the end time and duration here, in which case `select_interval()` must be called before the query is executed.

- **days** (`integer`) – The number of days in the time interval.
- **hours** (`integer`) – The number of hours in the time interval.
- **minutes** (`integer`) – The number of minutes in the time interval.

Raises `ValueError` if `end_time` is specified but `days`, `hours`, and `minutes` are all zero. If you really want to specify a point in time, use `select_interval()`.

align (*per_series_aligner, seconds=0, minutes=0, hours=0*)

Copy the query and add temporal alignment.

If `per_series_aligner` is not `Aligner.ALIGN_NONE`, each time series will contain data points only on the period boundaries.

Example:

```
query = query.align(Aligner.ALIGN_MEAN, minutes=5)
```

It is also possible to specify the aligner as a literal string:

```
query = query.align('ALIGN_MEAN', minutes=5)
```

Parameters

- **per_series_aligner** (*string*) – The approach to be used to align individual time series. For example: `Aligner.ALIGN_MEAN`. See *Aligner* and the descriptions of the [supported aligners](#).
- **seconds** (*integer*) – The number of seconds in the alignment period.
- **minutes** (*integer*) – The number of minutes in the alignment period.
- **hours** (*integer*) – The number of hours in the alignment period.

Return type *Query*

Returns The new query object.

as_dataframe (*label=None, labels=None*)

Return all the selected time series as a `pandas` dataframe.

Note: Use of this method requires that you have `pandas` installed.

Examples:

```
# Generate a dataframe with a multi-level column header including
# the resource type and all available resource and metric labels.
# This can be useful for seeing what labels are available.
dataframe = query.as_dataframe()

# Generate a dataframe using a particular label for the column
# names.
dataframe = query.as_dataframe(label='instance_name')

# Generate a dataframe with a multi-level column header.
dataframe = query.as_dataframe(labels=['zone', 'instance_name'])

# Generate a dataframe with a multi-level column header, assuming
# the metric is issued by more than one type of resource.
dataframe = query.as_dataframe(
    labels=['resource_type', 'instance_id'])
```

Parameters

- **label** (*string or None*) – The label name to use for the dataframe header. This can be the name of a resource label or metric label (e.g., "instance_name"), or the string "resource_type".
- **labels** (*list of strings, or None*) – A list or tuple of label names to use for the dataframe header. If more than one label name is provided, the resulting dataframe will have a multi-level column header. Providing values for both `label` and `labels` is an error.

Return type `pandas.DataFrame`

Returns A dataframe where each column represents one time series.

copy()

Copy the query object.

Return type `Query`

Returns The new query object.

filter

The filter string.

This is constructed from the metric type, the resource type, and selectors for the group ID, monitored projects, resource labels, and metric labels.

iter (*headers_only=False, page_size=None*)

Yield all time series objects selected by the query.

The generator returned iterates over `TimeSeries` objects containing points ordered from oldest to newest.

Note that the `Query` object itself is an iterable, such that the following are equivalent:

```
for timeseries in query:
    ...

for timeseries in query.iter():
    ...
```

Parameters

- **headers_only** (*boolean*) – Whether to omit the point data from the time series objects.
- **page_size** (*integer or None*) – An optional positive number specifying the maximum number of points to return per page. This can be used to control how far the iterator reads ahead.

Raises `ValueError` if the query time interval has not been specified.

metric_type

The metric type name.

reduce (*cross_series_reducer, *group_by_fields*)

Copy the query and add cross-series reduction.

Cross-series reduction combines time series by aggregating their data points.

For example, you could request an aggregated time series for each combination of project and zone as follows:

```
query = query.reduce(Reducer.REDUCE_MEAN,
                    'resource.project_id', 'resource.zone')
```

Parameters

- **cross_series_reducer** (*string*) – The approach to be used to combine time series. For example: `Reducer.REDUCE_MEAN`. See `Reducer` and the descriptions of the supported reducers.

- **group_by_fields** (*strings*) – Fields to be preserved by the reduction. For example, specifying just "resource.zone" will result in one time series per zone. The default is to aggregate all of the time series into just one.

Return type *Query*

Returns The new query object.

select_group (*group_id*)

Copy the query and add filtering by group.

Example:

```
query = query.select_group('1234567')
```

Parameters **group_id** (*string*) – The ID of a group to filter by.

Return type *Query*

Returns The new query object.

select_interval (*end_time*, *start_time=None*)

Copy the query and set the query time interval.

Example:

```
import datetime

now = datetime.datetime.utcnow()
query = query.select_interval(
    end_time=now,
    start_time=now - datetime.timedelta(minutes=5))
```

As a convenience, you can alternatively specify the end time and an interval duration when you create the query initially.

Parameters

- **end_time** (*datetime.datetime*) – The end time (inclusive) of the time interval for which results should be returned, as a datetime object.
- **start_time** (*datetime.datetime* or *None*) – The start time (exclusive) of the time interval for which results should be returned, as a datetime object. If not specified, the interval is a point in time.

Return type *Query*

Returns The new query object.

select_metrics (**args*, ***kwargs*)

Copy the query and add filtering by metric labels.

Examples:

```
query = query.select_metrics(instance_name='myinstance')
query = query.select_metrics(instance_name_prefix='mycluster-')
```

A keyword argument `<label>=<value>` ordinarily generates a filter expression of the form:

```
metric.label.<label> = "<value>"
```

However, by adding "`_prefix`" or "`_suffix`" to the keyword, you can specify a partial match.

`<label>_prefix=<value>` generates:

```
metric.label.<label> = starts_with("<value>")
```

`<label>_suffix=<value>` generates:

```
metric.label.<label> = ends_with("<value>")
```

If the label's value type is INT64, a similar notation can be used to express inequalities:

`<label>_less=<value>` generates:

```
metric.label.<label> < <value>
```

`<label>_lessequal=<value>` generates:

```
metric.label.<label> <= <value>
```

`<label>_greater=<value>` generates:

```
metric.label.<label> > <value>
```

`<label>_greaterequal=<value>` generates:

```
metric.label.<label> >= <value>
```

Parameters

- **args** (*tuple*) – Raw filter expression strings to include in the conjunction. If just one is provided and no keyword arguments are provided, it can be a disjunction.
- **kwargs** (*dict*) – Label filters to include in the conjunction as described above.

Return type *Query*

Returns The new query object.

select_projects (*args)

Copy the query and add filtering by monitored projects.

This is only useful if the target project represents a Stackdriver account containing the specified monitored projects.

Examples:

```
query = query.select_projects('project-1')
query = query.select_projects('project-1', 'project-2')
```

Parameters **args** (*tuple*) – Project IDs limiting the resources to be included in the query.

Return type *Query*

Returns The new query object.

select_resources (*args, **kwargs)

Copy the query and add filtering by resource labels.

Examples:

```
query = query.select_resources(zone='us-central1-a')
query = query.select_resources(zone_prefix='europe-')
query = query.select_resources(resource_type='gce_instance')
```

A keyword argument `<label>=<value>` ordinarily generates a filter expression of the form:

```
resource.label.<label> = "<value>"
```

However, by adding `"_prefix"` or `"_suffix"` to the keyword, you can specify a partial match.

`<label>_prefix=<value>` generates:

```
resource.label.<label> = starts_with("<value>")
```

`<label>_suffix=<value>` generates:

```
resource.label.<label> = ends_with("<value>")
```

As a special case, `"resource_type"` is treated as a special pseudo-label corresponding to the filter object `resource.type`. For example, `resource_type=<value>` generates:

```
resource.type = "<value>"
```

See the [defined resource types](#).

Note: The label `"instance_name"` is a metric label, not a resource label. You would filter on it using `select_metrics(instance_name=...)`.

Parameters

- **args** (*tuple*) – Raw filter expression strings to include in the conjunction. If just one is provided and no keyword arguments are provided, it can be a disjunction.
- **kwargs** (*dict*) – Label filters to include in the conjunction as described above.

Return type *Query*

Returns The new query object.

class `google.cloud.monitoring.query.Reducer`

Bases: `object`

Allowed values for the [supported reducers](#).

Time Series

Time series for the [Google Stackdriver Monitoring API \(V3\)](#).

Features intentionally omitted from this first version of the client library:

- Writing time series.
- Natural representation of distribution values.

class `google.cloud.monitoring.timeseries.Point`
 Bases: `google.cloud.monitoring.timeseries.Point`

A single point in a time series.

Parameters

- **end_time** (*string*) – The end time in RFC3339 UTC “Zulu” format.
- **start_time** (*string or None*) – An optional start time in RFC3339 UTC “Zulu” format.
- **value** (*object*) – The metric value. This can be a scalar or a distribution.

class `google.cloud.monitoring.timeseries.TimeSeries`
 Bases: `google.cloud.monitoring.timeseries.TimeSeries`

A single time series of metric values.

The preferred way to construct a `TimeSeries` object is using the `time_series()` factory method of the `Client` class.

Parameters

- **metric** (*Metric*) – A metric object.
- **resource** (*Resource*) – A resource object.
- **metric_kind** (*string*) – The kind of measurement: `MetricKind.GAUGE`, `MetricKind.DELTA`, or `MetricKind.CUMULATIVE`. See [MetricKind](#).
- **value_type** (*string*) – The value type of the metric: `ValueType.BOOL`, `ValueType.INT64`, `ValueType.DOUBLE`, `ValueType.STRING`, or `ValueType.DISTRIBUTION`. See [ValueType](#).
- **points** (list of *Point*) – A list of point objects.

header (*points=None*)

Copy everything but the point data.

Parameters **points** (list of *Point*, or None) – An optional point list.

Return type *TimeSeries*

Returns The new time series object.

labels

A single dictionary with values for all the labels.

This combines `resource.labels` and `metric.labels` and also adds "resource_type".

Label Descriptors

Label Descriptors for the Stackdriver Monitoring API (V3).

class `google.cloud.monitoring.label.LabelDescriptor` (*key*, *value_type*='STRING', *description*='')

Bases: `object`

Schema specification and documentation for a single label.

Parameters

- **key** (*string*) – The name of the label.
- **value_type** (*string*) – The type of the label. It must be one of `LabelValueType.STRING`, `LabelValueType.BOOL`, or `LabelValueType.INT64`. See [LabelValueType](#).
- **description** (*string*) – A human-readable description for the label.

class `google.cloud.monitoring.label.LabelValueType`

Bases: `object`

Allowed values for the type of a label.

Using the API

With [Google Translate](#), you can dynamically translate text between thousands of language pairs. The Google Translate API lets websites and programs integrate with Google Translate programmatically. Google Translate API is available as a paid service. See the [Pricing](#) and [FAQ](#) pages for details.

76.1 Authentication / Configuration

- Use *Client* objects to configure your applications.
- *Client* objects hold both a key and a connection to the Translate service.
- **An API key is required for Translate.** See [Identifying your application to Google](#) for details. This is significantly different than the other clients in `google-cloud-python`.

76.2 Methods

To create a client:

```
>>> from google.cloud import translate
>>> client = translate.Client('my-api-key')
```

By default, the client targets English when doing detections and translations, but a non-default value can be used as well:

```
>>> from google.cloud import translate
>>> client = translate.Client('my-api-key', target_language='es')
```

The Google Translate API has three supported methods, and they map to three methods on a client: `get_languages()`, `detect_language()` and `translate()`.

To get a list of languages supported by Google Translate

```
>>> from google.cloud import translate
>>> client = translate.Client('my-api-key')
>>> client.get_languages()
[
  {
    'language': 'af',
    'name': 'Afrikaans',
  },
]
```

```
    ...  
  ]
```

To detect the language that some given text is written in:

```
>>> from google.cloud import translate  
>>> client = translate.Client('my-api-key')  
>>> client.detect_language(['Me llamo', 'I am'])  
[  
  {  
    'confidence': 0.25830904,  
    'input': 'Me llamo',  
    'language': 'es',  
  }, {  
    'confidence': 0.17112699,  
    'input': 'I am',  
    'language': 'en',  
  },  
]
```

The `confidence` value is an optional floating point value between 0 and 1. The closer this value is to 1, the higher the confidence level for the language detection. This member is not always available.

To translate text:

```
>>> from google.cloud import translate  
>>> client = translate.Client('my-api-key')  
>>> client.translate('koszula')  
{  
  'translatedText': 'shirt',  
  'detectedSourceLanguage': 'pl',  
  'input': 'koszula',  
}
```

or to use a non-default target language:

```
>>> from google.cloud import translate  
>>> client = translate.Client('my-api-key')  
>>> client.translate(['Me llamo Jeff', 'My name is Jeff'],  
...                  target_language='de')  
[  
  {  
    'translatedText': 'Mein Name ist Jeff',  
    'detectedSourceLanguage': 'es',  
    'input': 'Me llamo Jeff',  
  }, {  
    'translatedText': 'Mein Name ist Jeff',  
    'detectedSourceLanguage': 'en',  
    'input': 'My name is Jeff',  
  },  
]
```

Translate Client

Client for interacting with the Google Cloud Translate API.

class `google.cloud.translate.client.Client` (*api_key*, *http=None*, *target_language='en'*)
 Bases: `object`

Client to bundle configuration needed for API requests.

Parameters

- **api_key** (*str*) – The key used to send with requests as a query parameter.
- **http** (`httplib2.Http` or class that defines `request()`) – (Optional) HTTP object to make requests. If not passed, an `httplib.Http` object is created.
- **target_language** (*str*) – (Optional) The target language used for translations and language names. (Defaults to `ENGLISH_ISO_639`.)

detect_language (*values*)

Detect the language of a string or list of strings.

See: <https://cloud.google.com/translate/v2/detecting-language-with-rest>

Parameters **values** (*str or list*) – String or list of strings that will have language detected.

Return type `str or list`

Returns

A list of dictionaries for each queried value. Each dictionary typically contains three keys

- `confidence`: The confidence in language detection, a float between 0 and 1.
- `input`: The corresponding input value.
- `language`: The detected language (as an ISO 639-1 language code).

though the key `confidence` may not always be present.

If only a single value is passed, then only a single dictionary will be returned.

Raises `ValueError` if the number of detections is not equal to the number of values.
`ValueError` if a value produces a list of detections with 0 or multiple results in it.

get_languages (*target_language=None*)

Get list of supported languages for translation.

Response

See: <https://cloud.google.com/translate/v2/discovering-supported-languages-with-rest>

Parameters `target_language` (*str*) – (Optional) The language used to localize returned language names. Defaults to the target language on the current client.

Return type `list`

Returns List of dictionaries. Each dictionary contains a supported ISO 639-1 language code (using the dictionary key `language`). If `target_language` is passed, each dictionary will also contain the name of each supported language (localized to the target language).

translate (*values*, *target_language=None*, *format_=None*, *source_language=None*, *customization_ids=()*)

Translate a string or list of strings.

See: <https://cloud.google.com/translate/v2/translating-text-with-rest>

Parameters

- **values** (*str or list*) – String or list of strings to translate.
- **target_language** (*str*) – The language to translate results into. This is required by the API and defaults to the target language of the current instance.
- **format** (*str*) – (Optional) One of `text` or `html`, to specify if the input text is plain text or HTML.
- **source_language** (*str*) – (Optional) The language of the text to be translated.
- **customization_ids** (*str or list*) – (Optional) ID or list of customization IDs for translation. Sets the `cid` parameter in the query.

Return type `str or list list`

Returns

A list of dictionaries for each queried value. Each dictionary typically contains three keys (though not all will be present in all cases)

- `detectedSourceLanguage`: The detected language (as an ISO 639-1 language code) of the text.
- `translatedText`: The translation of the text into the target language.
- `input`: The corresponding input value.

If only a single value is passed, then only a single dictionary will be returned.

Raises `ValueError` if the number of values and translations differ.

`google.cloud.translate.client.ENGLISH_ISO_639 = 'en'`
ISO 639-1 language code for English.

77.1 Connection

Create / interact with Google Cloud Translate connections.

class `google.cloud.translate.connection.Connection` (*credentials=None*, *http=None*)
Bases: `google.cloud.connection.JSONConnection`

A connection to Google Cloud Translate 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}/language/translate/{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.

Using the Vision API

78.1 Authentication and Configuration

- For an overview of authentication in `google-cloud-python`, see [Authentication](#).
- In addition to any authentication configuration, you should also set the `GOOGLE_CLOUD_PROJECT` environment variable for the project you'd like to interact with. If the `GOOGLE_CLOUD_PROJECT` environment variable is not present, the project ID from JSON file credentials is used.

If you are using Google App Engine or Google Compute Engine this will be detected automatically.

- After configuring your environment, create a `Client`

```
>>> from google.cloud import vision
>>> client = vision.Client()
```

or pass in credentials and project explicitly

```
>>> from google.cloud import vision
>>> client = vision.Client(project='my-project', credentials=creds)
```

78.2 Annotating an Image

78.2.1 Annotate a single image

```
>>> import io
>>> from google.cloud import vision
>>> client = vision.Client()
>>> with io.open('./image.png', 'rb') as image_file:
...     image = client.image(content=image_file.read())
>>> faces = image.detect_faces(limit=10)
>>> faces[0].landmarks.left_eye.position.x_coordinate
... 1004.8003
```

78.2.2 Annotate multiple images

```
>>> import io
>>> from google.cloud import vision
>>> client = vision.Client()
>>> with io.open('./image.png', 'rb') as image_file:
...     image_one = client.image(content=image_file.read())
>>> image_two = client.image(source_uri='gs://my-storage-bucket/image.jpg')
>>> with client.batch():
...     labels = image_one.detect_labels()
...     faces = image_two.detect_faces(limit=10)
```

78.2.3 No results returned

Failing annotations return no results for the feature type requested.

```
>>> from google.cloud import vision
>>> client = vision.Client()
>>> image = client.image(source_uri='gs://my-storage-bucket/image.jpg')
>>> logos = image.detect_logos(limit=10)
>>> logos
[]
```

78.2.4 Manual Detection

You can call the detection method manually.

```
>>> from google.cloud import vision
>>> from google.cloud.vision.image import Feature
>>> from google.cloud.vision.image import FeatureTypes
>>> client = vision.Client()
>>> image = client.image(source_uri='gs://my-test-bucket/image.jpg')
>>> features = [Feature(FeatureTypes.FACE_DETECTION, 5),
...             Feature(FeatureTypes.LOGO_DETECTION, 3)]
>>> annotations = image.detect(features)
```

78.2.5 Face Detection

Detecting a face or faces in an image. For a list of the possible facial landmarks see: https://cloud.google.com/vision/reference/rest/v1/images/annotate#type_1

```
>>> from google.cloud import vision
>>> client = vision.Client()
>>> image = client.image(source_uri='gs://my-test-bucket/image.jpg')
>>> faces = image.detect_faces(limit=10)
>>> faces[0].landmarks.left_eye.landmark_type
'LEFT_EYE'
>>> faces[0].landmarks.left_eye.position.x_coordinate
1301.2404
>>> faces[0].detection_confidence
0.9863683
```

```
>>> faces[0].joy_likelihood
0.54453093
>>> faces[0].anger_likelihood
0.02545464
```

78.2.6 Label Detection

Image labels are a way to help categorize the contents of an image. If you have an image with a car, person and a dog it, label detection will attempt to identify those objects.

```
>>> from google.cloud import vision
>>> client = vision.Client()
>>> image = client.image(source_uri='gs://my-storage-bucket/image.jpg')
>>> labels = image.detect_labels(limit=3)
>>> labels[0].description
'automobile'
>>> labels[0].score
0.9863683
```

78.2.7 Landmark Detection

The API will attempt to detect landmarks such as Mount Rushmore and the Sydney Opera House. The API will also provide their known geographical locations if available.

```
>>> from google.cloud import vision
>>> client = vision.Client()
>>> image = client.image('./image.jpg')
>>> landmarks = image.detect_landmarks()
>>> landmarks[0].description
'Sydney Opera House'
>>> landmarks[0].locations[0].latitude
-33.857123
>>> landmarks[0].locations[0].longitude
151.213921
>>> landmarks[0].bounding_poly.vertices[0].x_coordinate
78
>>> landmarks[0].bounding_poly.vertices[0].y_coordinate
162
```

78.2.8 Logo Detection

Google Vision can also attempt to detect company and brand logos in images.

```
>>> from google.cloud import vision
>>> client = vision.Client()
>>> image = client.image('./image.jpg')
>>> logos = image.detect_logos(limit=1)
>>> results.logos[0].description
'Google'
>>> logos[0].score
0.9795432
```

```
>>> logos[0].bounding_poly.vertices[0].x_coordinate
78
>>> logos[0].bounding_poly.vertices[0].y_coordinate
62
```

78.2.9 Safe Search Detection

Detecting safe search properties of an image.

```
>>> from google.cloud import vision
>>> client = vision.Client()
>>> image = client.image('./image.jpg')
>>> safe_search = image.detect_safe_search()
>>> safe_search.adult
'VERY_UNLIKELY'
>>> safe_search.medical
'UNLIKELY'
```

78.2.10 Text Detection

Detecting text with ORC from an image.

```
>>> from google.cloud import vision
>>> client = vision.Client()
>>> image = client.image('./image.jpg')
>>> texts = image.detect_text()
>>> texts[0].locale
'en'
>>> texts[0].description
'some text in the image'
>>> texts[1].description
'some other text in the image'
```

78.2.11 Image Properties

Detecting image color properties.

```
>>> from google.cloud import vision
>>> client = vision.Client()
>>> image = client.image('./image.jpg')
>>> colors = image.detect_properties()
>>> colors[0].red
244
>>> colors[0].blue
134
>>> colors[0].score
0.65519291
>>> colors[0].pixel_fraction
0.758658
```

Vision Client

79.1 Client

Client for interacting with the Google Cloud Vision API.

class `google.cloud.vision.client.Client` (*project=None, credentials=None, http=None*)
 Bases: `google.cloud.client.JSONClient`

Client to bundle configuration needed for API requests.

Parameters

- **project** (*str*) – 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.

annotate (*image, features*)

Annotate an image to discover it's attributes.

Parameters

- **image** (*str*) – A string which can be a URL, a Google Cloud Storage path, or a byte stream of the image.
- **features** (list of *Feature*) – The type of detection that the Vision API should use to determine image attributes. Pricing is based on the number of Feature Types.

See: <https://cloud.google.com/vision/docs/pricing>

Return type `dict`

Returns List of annotations.

image (*content=None, source_uri=None*)

Get instance of Image using current client.

Parameters

- **content** (*bytes*) – Byte stream of an image.

- **source_uri** (*str*) – Google Cloud Storage URI of image.

Return type *Image*

Returns Image instance with the current client attached.

class `google.cloud.vision.client.VisionRequest` (*image, features*)

Bases: `object`

Request container with image and features information to annotate.

Parameters

- **features** (list of `Feature`.) – The features that dictate which annotations to run.
- **image** (*bytes*) – Either Google Cloud Storage URI or raw byte stream of image.

as_dict ()

Dictionary representation of Image.

features

List of Feature objects.

image

Image object containing image content.

79.2 Connection

Create / interact with Google Cloud Vision connections.

class `google.cloud.vision.connection.Connection` (*credentials=None, http=None*)

Bases: `google.cloud.connection.JSONConnection`

A connection to Google Cloud Vision 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://vision.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/cloud-platform',)`

The scopes required for authenticating as a Cloud Vision consumer.

Vision Image Properties

80.1 Image Properties Annotation

Image properties class representation derived from Vision API response.

class `google.cloud.vision.color.Color` (*red, green, blue, alpha*)
Bases: `object`

Representation of RGBA color information.

Parameters

- **red** (*int*) – The amount of red in the color as a value in the interval [0, 255].
- **green** (*int*) – The amount of green in the color as a value in the interval [0, 255].
- **blue** (*int*) – The amount of blue in the color as a value in the interval [0, 255].
- **alpha** (*float*) – The fraction of this color that should be applied to the pixel.

alpha

Alpha transparency level.

Return type `float`

Returns Alpha transparency level.

blue

Blue component of the color.

Return type `int`

Returns Blue RGB value.

classmethod `from_api_repr` (*response*)

Factory: construct a `Color` from a Vision API response.

Parameters **response** (*dict*) – Color from API Response.

Return type `Color`

Returns Instance of `Color`.

green

Green component of the color.

Return type `int`

Returns Green RGB value.

red

Red component of the color.

Return type `int`

Returns Red RGB value.

class `google.cloud.vision.color.ColorInformation` (*color, score, pixel_fraction*)

Bases: `object`

Representation of color information from API response.

Parameters

- **color** (*Color*) – RGB components of the color.
- **score** (*float*) – Image-specific score for this color. Value in range [0, 1].
- **pixel_fraction** (*float*) – Stores the fraction of pixels the color occupies in the image. Value in range [0, 1].

color

RGB components of the color.

Return type `Color`

Returns Instance of `Color`.

classmethod `from_api_repr` (*response*)

Factory: construct `ColorInformation` for a color found.

Parameters **response** (*dict*) – Color data with extra meta information.

Return type `ColorInformation`

Returns Instance of `ColorInformation`.

pixel_fraction

Stores the fraction of pixels the color occupies in the image.

Return type `float`

Returns Pixel fraction value in range [0, 1].

score

Image-specific score for this color. Value in range [0, 1].

Return type `float`

Returns Image score for this color.

class `google.cloud.vision.color.ImagePropertiesAnnotation` (*colors*)

Bases: `object`

Representation of image properties

Parameters **colors** (*list*) – List of `ColorInformation`.

colors

Colors in an image.

Return type list of `ColorInformation`

Returns Populated list of `ColorInformation`.

classmethod `from_api_repr` (*response*)

Factory: construct `ImagePropertiesAnnotation` from a response.

Parameters `response` (*dict*) – Dictionary response from Vision API with image properties data.

Return type *ImagePropertiesAnnotation*.

Returns Populated instance of *ImagePropertiesAnnotation*.

Vision Entity

81.1 Entity

Entity class for holding information returned from annotating an image.

class `google.cloud.vision.entity.EntityAnnotation` (*bounds, description, locale, locations, mid, score*)

Bases: `object`

Representation of an entity returned from the Vision API.

Parameters

- **bounds** (*dict*) – Dictionary of boundary information of detected entity.
- **description** (*str*) – Description of entity detected in an image.
- **locale** (*str*) – The language code for the locale in which the entity textual description (next field) is expressed.
- **locations** (list of *LocationInformation*.) – List of *LocationInformation* instances.
- **mid** (*str*) – Opaque entity ID.
- **score** (*float*) – Overall score of the result. Range [0, 1].

bounds

Bounding polygon of detected image feature.

Return type *Bounds*

Returns Instance of *Bounds* with populated vertices.

description

Description of feature detected in image.

Return type *str*

Returns String description of feature detected in image.

classmethod `from_api_repr` (*response*)

Factory: construct entity from Vision API response.

Parameters **response** (*dict*) – Dictionary response from Vision API with entity data.

Return type *EntityAnnotation*

Returns Instance of *EntityAnnotation*.

locale

The language code for text discovered in an image.

Return type `str`

Returns String language code of text found in the image.

locations

Location coordinates landmarks detected.

Return type `LocationInformation`

Returns `LocationInformation` populated with latitude and longitude of object detected in an image.

mid

MID of feature detected in image.

Return type `str`

Returns String MID of feature detected in image.

score

Overall score of the result. Range [0, 1].

Return type `float`

Returns Overall score of the result. Range [0, 1].

Vision Feature

82.1 Feature

Feature representing various types of annotations.

class `google.cloud.vision.feature.Feature` (*feature_type*, *max_results=1*)
 Bases: `object`

Feature object specifying the annotation type and maximum results.

Parameters

- **feature_type** (*str*) – String representation of `FeatureType`.
- **max_results** (*int*) – Number of results to return for the specified feature type.

See: <https://cloud.google.com/vision/reference/rest/v1/images/annotate#Feature>

as_dict ()
 Generate dictionary for Feature request format.

Return type `dict`

Returns Dictionary representation of a `FeatureType`.

feature_type
 “Feature type string.

Return type `FeatureTypes`

Returns Instance of `FeatureTypes`

max_results
 Maximum number of results for feature type.

Return type `int`

Returns Maximum results to be returned.

class `google.cloud.vision.feature.FeatureTypes`
 Bases: `object`

Feature Types to indicate which annotations to perform.

See: <https://cloud.google.com/vision/reference/rest/v1/images/annotate#Type>

FACE_DETECTION = ‘FACE_DETECTION’

IMAGE_PROPERTIES = ‘IMAGE_PROPERTIES’

LABEL_DETECTION = 'LABEL_DETECTION'

LANDMARK_DETECTION = 'LANDMARK_DETECTION'

LOGO_DETECTION = 'LOGO_DETECTION'

SAFE_SEARCH_DETECTION = 'SAFE_SEARCH_DETECTION'

TEXT_DETECTION = 'TEXT_DETECTION'

83.1 Face

Face class representing the Vision API's face detection response.

class `google.cloud.vision.face.Angles` (*roll, pan, tilt*)
Bases: `object`

Angles representing the positions of a face.

classmethod `from_api_repr` (*response*)
Factory: construct the angles from an Vision API response.

Return type `Angles`

Returns An `Angles` instance with data parsed from *response*.

pan

Pan angle of face.

Return type `float`

Returns Pan angle of face in degrees.

roll

Roll angle of face.

Return type `float`

Returns Roll angle of face in degrees.

tilt

Tilt angle of face.

Return type `float`

Returns Tilt angle of face in degrees.

class `google.cloud.vision.face.Bounds` (*vertices*)
Bases: `google.cloud.vision.geometry.BoundsBase`

The bounding polygon of the entire face.

class `google.cloud.vision.face.Emotions` (*joy_likelihood, sorrow_likelihood, surprise_likelihood, anger_likelihood*)
Bases: `object`

Emotions displayed by the face detected in an image.

anger_likelihood

Likelihood of anger in detected face.

Return type `str`

Returns String derived from `Likelihood`.

classmethod from_api_repr (*response*)

Factory: construct *Emotions* from Vision API response.

Parameters **response** (*dict*) – Response dictionary representing a face.

Return type `Emotions`

Returns Populated instance of *Emotions*.

joy_likelihood

Likelihood of joy in detected face.

Return type `str`

Returns String derived from `Likelihood`.

sorrow_likelihood

Likelihood of sorrow in detected face.

Return type `str`

Returns String derived from `Likelihood`.

surprise_likelihood

Likelihood of surprise in detected face.

Return type `str`

Returns String derived from `Likelihood`.

class `google.cloud.vision.face.FDBounds` (*vertices*)

Bases: `google.cloud.vision.geometry.BoundsBase`

The bounding polygon of just the skin portion of the face.

class `google.cloud.vision.face.Face` (*angles, bounds, detection_confidence, emotions, fd_bounds, headwear_likelihood, image_properties, landmarks, landmarking_confidence*)

Bases: `object`

Representation of a face found by the Vision API

angles

Accessor to the pan, tilt and roll angles of a `Face`.

Return type `Angles`

Returns Pan, tilt and roll angles of the detected face.

bounds

Accessor to the bounding poly information of the detected face.

Return type `Bounds`

Returns An instance of `Bounds` which has a list of vertices.

detection_confidence

Face detection confidence score determined by the Vision API.

Return type `float`

Returns Float representation of confidence ranging from 0 to 1.

emotions

Accessor to the possible emotions expressed in the detected face.

Return type *Emotions*

Returns An instance of *Emotions* with joy, sorrow, anger, surprise likelihood.

fd_bounds

Accessor to the skin area bounding poly of the detected face.

Return type *FDBounds*

Returns An instance of *FDBounds* which has a list of vertices.

classmethod from_api_repr (*response*)

Factory: construct an instance of a *Face* from an API response

Parameters **response** (*dict*) – Face annotation dict returned from the Vision API.

Return type *Face*

Returns A instance of *Face* with data parsed from *response*.

headwear_likelihood

Headwear likelihood.

Return type *Likelihood*

Returns String representing the likelihood based on *Likelihood*

image_properties

Image properties from imaged used in face detection.

Return type *FaceImageProperties*

Returns *FaceImageProperties* object with image properties.

landmarking_confidence

Landmarking confidence score determined by the Vision API.

Return type *float*

Returns Float representing the confidence of the Vision API in determining the landmarks on a face.

landmarks

Accessor to the facial landmarks detected in a face.

Return type *Landmarks*

Returns *Landmarks* object with facial landmarks as properties.

class `google.cloud.vision.face.FaceImageProperties` (*blurred_likelihood*, *underexposed_likelihood*)

Bases: *object*

A representation of the image properties from face detection.

blurred_likelihood

Likelihood of the image being blurred.

Return type *str*

Returns String representation derived from *Position*.

classmethod `from_api_repr` (*response*)

Factory: construct image properties from image.

Return type `FaceImageProperties`

Returns Instance populated with image property data.

underexposed_likelihood

Likelihood that the image used for detection was underexposed.

Return type `str`

Returns String representation derived from `Position`.

class `google.cloud.vision.face.FaceLandmarkTypes`

Bases: `object`

A representation of the face detection landmark types.

See: https://cloud.google.com/vision/reference/rest/v1/images/annotate#Type_1

`CHIN_GNATHION = 'CHIN_GNATHION'`

`CHIN_LEFT_GONION = 'CHIN_LEFT_GONION'`

`CHIN_RIGHT_GONION = 'CHIN_RIGHT_GONION'`

`FOREHEAD_GLABELLA = 'FOREHEAD_GLABELLA'`

`LEFT_EAR_TRAGION = 'LEFT_EAR_TRAGION'`

`LEFT_EYE = 'LEFT_EYE'`

`LEFT_EYEBROW_UPPER_MIDPOINT = 'LEFT_EYEBROW_UPPER_MIDPOINT'`

`LEFT_EYE_BOTTOM_BOUNDARY = 'LEFT_EYE_BOTTOM_BOUNDARY'`

`LEFT_EYE_LEFT_CORNER = 'LEFT_EYE_LEFT_CORNER'`

`LEFT_EYE_PUPIL = 'LEFT_EYE_PUPIL'`

`LEFT_EYE_RIGHT_CORNER = 'LEFT_EYE_RIGHT_CORNER'`

`LEFT_EYE_TOP_BOUNDARY = 'LEFT_EYE_TOP_BOUNDARY'`

`LEFT_OF_LEFT_EYEBROW = 'LEFT_OF_LEFT_EYEBROW'`

`LEFT_OF_RIGHT_EYEBROW = 'LEFT_OF_RIGHT_EYEBROW'`

`LOWER_LIP = 'LOWER_LIP'`

`MIDPOINT_BETWEEN_EYES = 'MIDPOINT_BETWEEN_EYES'`

`MOUTH_CENTER = 'MOUTH_CENTER'`

`MOUTH_LEFT = 'MOUTH_LEFT'`

`MOUTH_RIGHT = 'MOUTH_RIGHT'`

`NOSE_BOTTOM_CENTER = 'NOSE_BOTTOM_CENTER'`

`NOSE_BOTTOM_LEFT = 'NOSE_BOTTOM_LEFT'`

`NOSE_BOTTOM_RIGHT = 'NOSE_BOTTOM_RIGHT'`

`NOSE_TIP = 'NOSE_TIP'`

`RIGHT_EAR_TRAGION = 'RIGHT_EAR_TRAGION'`

`RIGHT_EYE = 'RIGHT_EYE'`

```

RIGHT_EYEBROW_UPPER_MIDPOINT = 'RIGHT_EYEBROW_UPPER_MIDPOINT'
RIGHT_EYE_BOTTOM_BOUNDARY = 'RIGHT_EYE_BOTTOM_BOUNDARY'
RIGHT_EYE_LEFT_CORNER = 'RIGHT_EYE_LEFT_CORNER'
RIGHT_EYE_PUPIL = 'RIGHT_EYE_PUPIL'
RIGHT_EYE_RIGHT_CORNER = 'RIGHT_EYE_RIGHT_CORNER'
RIGHT_EYE_TOP_BOUNDARY = 'RIGHT_EYE_TOP_BOUNDARY'
RIGHT_OF_LEFT_EYEBROW = 'RIGHT_OF_LEFT_EYEBROW'
RIGHT_OF_RIGHT_EYEBROW = 'RIGHT_OF_RIGHT_EYEBROW'
UNKNOWN_LANDMARK = 'UNKNOWN_LANDMARK'
UPPER_LIP = 'UPPER_LIP'

```

class `google.cloud.vision.face.Landmark` (*position, landmark_type*)

Bases: `object`

A face-specific landmark (for example, a face feature, left eye).

classmethod `from_api_repr` (*response_landmark*)

Factory: construct an instance of a `Landmark` from a response.

Parameters `response_landmark` (*dict*) – Landmark representation from Vision API.

Return type `Landmark`

Returns Populated instance of `Landmark`.

landmark_type

Landmark type of facial feature.

Return type `str`

Returns String representation of facial landmark type.

position

Landmark position on face.

Return type `Position`

Returns Instance of `Position` with landmark coordinates.

class `google.cloud.vision.face.Landmarks` (*landmarks*)

Bases: `object`

Landmarks detected on a face represented as properties.

Vision Image

84.1 Image

Image represented by either a URI or byte stream.

class `google.cloud.vision.image.Image` (*client*, *content=None*, *source_uri=None*)

Bases: `object`

Image representation containing information to be annotate.

Parameters

- **content** (*bytes*) – Byte stream of an image.
- **source_uri** (*str*) – Google Cloud Storage URI of image.
- **client** (*Client*) – Instance of Vision client.

as_dict ()

Generate dictionary structure for request.

Return type `dict`

Returns Dictionary with source information for image.

content

Base64 encoded image content.

Return type `str`

Returns Base64 encoded image bytes.

detect_faces (*limit=10*)

Detect faces in image.

Parameters **limit** (*int*) – The number of faces to try and detect.

Return type `list`

Returns List of *Face*.

detect_labels (*limit=10*)

Detect labels that describe objects in an image.

Parameters **limit** (*int*) – The maximum number of labels to try and detect.

Return type `list`

Returns List of *EntityAnnotation*

detect_landmarks (*limit=10*)

Detect landmarks in an image.

Parameters **limit** (*int*) – The maximum number of landmarks to find.

Return type *list*

Returns List of *EntityAnnotation*.

detect_logos (*limit=10*)

Detect logos in an image.

Parameters **limit** (*int*) – The maximum number of logos to find.

Return type *list*

Returns List of *EntityAnnotation*.

detect_properties (*limit=10*)

Detect the color properties of an image.

Parameters **limit** (*int*) – The maximum number of image properties to find.

Return type *list*

Returns List of *ImagePropertiesAnnotation*.

detect_safe_search (*limit=10*)

Retrieve safe search properties from an image.

Parameters **limit** (*int*) – The number of faces to try and detect.

Return type *list*

Returns List of *SafeSearchAnnotation*.

detect_text (*limit=10*)

Detect text in an image.

Parameters **limit** (*int*) – The maximum instances of text to find.

Return type *list*

Returns List of *EntityAnnotation*.

source

Google Cloud Storage URI.

Return type *str*

Returns String of Google Cloud Storage URI.

84.2 Geometry

Geometry and other generic classes used by the Vision API.

class `google.cloud.vision.geometry.Bounds` (*vertices*)

Bases: `google.cloud.vision.geometry.BoundsBase`

A polygon boundry of the detected feature.

class `google.cloud.vision.geometry.BoundsBase` (*vertices*)

Bases: `object`

Base class for handling bounds with vertices.

Parameters `vertices` (list of `Vertex`) – List of vertices describing points on an image.

classmethod `from_api_repr` (`response_vertices`)

Factory: construct `BoundsBase` instance from Vision API response.

Parameters `response_vertices` (`dict`) – List of vertices.

Return type `BoundsBase` or `None`

Returns Instance of `BoundsBase` with populated vertices or `None`.

vertices

List of vertices.

Return type list of `Vertex`

Returns List of populated vertices.

class `google.cloud.vision.geometry.FDBounds` (`vertices`)

Bases: `google.cloud.vision.geometry.BoundsBase`

The bounding polygon of just the skin portion of the face.

class `google.cloud.vision.geometry.LocationInformation` (`latitude`, `longitude`)

Bases: `object`

Representation of location information returned by the Vision API.

Parameters

- **latitude** (`float`) – Latitude coordinate of geographical location.
- **longitude** (`float`) – Longitude coordinate of geographical location.

classmethod `from_api_repr` (`response`)

Factory: construct location information from Vision API response.

Parameters `response` (`dict`) – Dictionary response of locations.

Return type `LocationInformation`

Returns `LocationInformation` with populated latitude and longitude.

latitude

Latitude coordinate.

Return type `float`

Returns Latitude coordinate of location.

longitude

Longitude coordinate.

Return type `float`

Returns Longitude coordinate of location.

class `google.cloud.vision.geometry.Position` (`x_coordinate`, `y_coordinate`, `z_coordinate`)

Bases: `object`

A 3D position in the image.

See: <https://cloud.google.com/vision/reference/rest/v1/images/annotate#Position>

Parameters

- **x_coordinate** (`float`) – X position coordinate.
- **y_coordinate** (`float`) – Y position coordinate.

- **z_coordinate** (*float*) – Z position coordinate.

classmethod **from_api_repr** (*response_position*)

Factory: construct 3D position from API response.

Return type *Position*

Returns *Position* constructed with 3D points from API response.

x_coordinate

X position coordinate.

Return type *float*

Returns X position coordinate.

y_coordinate

Y position coordinate.

Return type *float*

Returns Y position coordinate.

z_coordinate

Z position coordinate.

Return type *float*

Returns Z position coordinate.

class `google.cloud.vision.geometry`.**Vertex** (*x_coordinate*, *y_coordinate*)

Bases: `object`

A vertex represents a 2D point in the image.

See: <https://cloud.google.com/vision/reference/rest/v1/images/annotate#Vertex>

Parameters

- **x_coordinate** (*float*) – X position coordinate.
- **y_coordinate** (*float*) – Y position coordinate.

x_coordinate

X position coordinate.

Return type *float*

Returns X position coordinate.

y_coordinate

Y position coordinate.

Return type *float*

Returns Y position coordinate.

84.3 Likelihood

Likelihood constants returned from Vision API.

class `google.cloud.vision.likelihood`.**Likelihood**

Bases: `object`

A representation of likelihood to give stable results across upgrades.

See: <https://cloud.google.com/vision/reference/rest/v1/images/annotate#likelihood>

LIKELY = 'LIKELY'

POSSIBLE = 'POSSIBLE'

UNKNOWN = 'UNKNOWN'

UNLIKELY = 'UNLIKELY'

VERY_LIKELY = 'VERY_LIKELY'

VERY_UNLIKELY = 'VERY_UNLIKELY'

Vision Safe Search

85.1 Safe Search Annotation

Safe search class for information returned from annotating an image.

```
class google.cloud.vision.safe.SafeSearchAnnotation (adult_likelihood, spoof_likelihood,
                                                    medical_likelihood, violence_likelihood)
```

Bases: `object`

Representation of a SafeSearchAnnotation.

Parameters

- **adult_likelihood** (*Likelihood*) – Likelihood that image contains adult material.
- **spoof_likelihood** (*Likelihood*) – Likelihood that image is a spoof.
- **medical_likelihood** (*Likelihood*) – Likelihood that image contains medical material.
- **violence_likelihood** (*Likelihood*) – Likelihood that image contains violence.

adult

Represents the adult contents likelihood for the image.

Return type *Likelihood*

Returns *Likelihood* of the image containing adult content.

classmethod **from_api_repr** (*response*)

Factory: construct SafeSearchAnnotation from Vision API response.

Parameters **response** (*dict*) – Dictionary response from Vision API with safe search data.

Return type *SafeSearchAnnotation*

Returns Instance of SafeSearchAnnotation.

medical

Likelihood this is a medical image.

Return type *Likelihood*

Returns The *Likelihood* that the image is medical in origin.

spoof

The likelihood that an obvious modification was made to the image.

Return type *Likelihood*

Returns The *Likelihood* that an obvious modification was made to the image's canonical version to make it appear funny or offensive.

violence

Likelihood that this image contains violence.

Return type *Likelihood*

Returns The *Likelihood* that the image contains violence.

Using the API

The [Google Natural Language API](#) can be used to reveal the structure and meaning of text via powerful machine learning models. You can use it to extract information about people, places, events and much more, mentioned in text documents, news articles or blog posts. You can use it to understand sentiment about your product on social media or parse intent from customer conversations happening in a call center or a messaging app. You can analyze text uploaded in your request or integrate with your document storage on Google Cloud Storage.

Warning: This is a Beta release of Google Cloud Natural Language API. This API is not intended for real-time usage in critical applications.

86.1 Client

Client objects provide a means to configure your application. Each instance holds an authenticated connection to the Natural Language service.

For an overview of authentication in `google-cloud-python`, see [Authentication](#).

Assuming your environment is set up as described in that document, create an instance of *Client*.

```
>>> from google.cloud import language
>>> client = language.Client()
```

By default the language is 'en-US' and the encoding is UTF-8. To over-ride these values:

```
>>> client = language.Client(language='es',
...                           encoding=language.Encoding.UTF16)
```

The encoding can be one of `Encoding.UTF8`, `Encoding.UTF16`, or `Encoding.UTF32`.

86.2 Methods

The Google Natural Language API has three supported methods

- `analyzeEntities`
- `analyzeSentiment`
- `annotateText`

and each method uses a `Document` for representing text. To create a `Document`,

```
>>> text_content = (
...     'Google, headquartered in Mountain View, unveiled the '
...     'new Android phone at the Consumer Electronic Show. '
...     'Sundar Pichai said in his keynote that users love '
...     'their new Android phones.')
>>> document = client.document_from_text(text_content)
```

By using `document_from_text()`, the document's type is plain text:

```
>>> document.doc_type == language.Document.PLAIN_TEXT
True
```

In addition, the document's language defaults to the language on the client

```
>>> document.language
'en-US'
>>> document.language == client.language
True
```

In addition, the `document_from_html()` factory can be used to create an HTML document. In this method and the from text method, the language can be over-riden:

```
>>> html_content = """\
... <html>
...   <head>
...     <title>El Tiempo de las Historias</title>
...   </head>
...   <body>
...     <p>La vaca saltó sobre la luna.</p>
...   </body>
... </html>
... """
>>> document = client.document_from_html(html_content,
...                                       language='es')
```

The language argument can be either ISO-639-1 or BCP-47 language codes; at the time, only English, Spanish, and Japanese are supported. However, the `analyzeSentiment` method only supports English text.

The document type (`doc_type`) value can be one of `Document.PLAIN_TEXT` or `Document.HTML`.

In addition to supplying the text / HTML content, a document can refer to content stored in Google Cloud Storage. We can use the `document_from_url()` method:

```
>>> gcs_url = 'gs://my-text-bucket/sentiment-me.txt'
>>> document = client.document_from_url(
...     gcs_url, doc_type=language.Document.HTML)
>>> document.gcs_url == gcs_url
True
>>> document.doc_type == language.Document.PLAIN_TEXT
True
```

The document type can be specified with the `doc_type` argument:

```
>>> document = client.document_from_url(
...     gcs_url, doc_type=language.Document.HTML)
```

86.3 Analyze Entities

The `analyze_entities()` method finds named entities (i.e. proper names) in the text and returns them as a list of `Entity` objects. Each entity has a corresponding type, salience (prominence), associated metadata and other properties.

```
>>> text_content = ("Michelangelo Caravaggio, Italian painter, is "
...                  "known for 'The Calling of Saint Matthew'.")
>>> document = client.document(text_content)
>>> entities = document.analyze_entities()
>>> for entity in entities:
...     print('=' * 20)
...     print('      name: %s' % (entity.name,))
...     print('      type: %s' % (entity.entity_type,))
...     print('wikipedia_url: %s' % (entity.wikipedia_url,))
...     print('      metadata: %s' % (entity.metadata,))
...     print('      salience: %s' % (entity.salience,))
=====
      name: Michelangelo Caravaggio
      type: PERSON
wikipedia_url: http://en.wikipedia.org/wiki/Caravaggio
      metadata: {}
      salience: 0.7615959
=====
      name: Italian
      type: LOCATION
wikipedia_url: http://en.wikipedia.org/wiki/Italy
      metadata: {}
      salience: 0.19960518
=====
      name: The Calling of Saint Matthew
      type: EVENT
wikipedia_url: http://en.wikipedia.org/wiki/The_Calling_of_St_Matthew_(Caravaggio)
      metadata: {}
      salience: 0.038798928
```

86.4 Analyze Sentiment

The `analyze_sentiment()` method analyzes the sentiment of the provided text and returns a `Sentiment`. Currently, this method only supports English text.

```
>>> text_content = "Jogging isn't very fun."
>>> document = client.document(text_content)
>>> sentiment = document.analyze_sentiment()
>>> print(sentiment.polarity)
-1
>>> print(sentiment.magnitude)
0.8
```

86.5 Annotate Text

The `annotate_text()` method analyzes a document and is intended for users who are familiar with machine learning and need in-depth text features to build upon.

The method returns a named tuple with four entries:

- `sentences`: A `list` of sentences in the text
- `tokens`: A `list` of `Token` object (e.g. words, punctuation)
- `sentiment`: The `Sentiment` of the text (as returned by `analyze_sentiment()`)
- `entities`: `list` of `Entity` objects extracted from the text (as returned by `analyze_entities()`)

By default `annotate_text()` has three arguments `include_syntax`, `include_entities` and `include_sentiment` which are all `True`. However, each of these `Features` can be selectively turned off by setting the corresponding arguments to `False`.

When `include_syntax=False`, `sentences` and `tokens` in the response is `None`. When `include_sentiment`, `sentiment` in the response is `None`. When `include_entities`, `entities` in the response is `None`.

```
>>> text_content = 'The cow jumped over the Moon.'
>>> document = client.document(text_content)
>>> annotations = document.annotate_text()
>>> # Sentences present if include_syntax=True
>>> print(annotations.sentences)
['The cow jumped over the Moon.']
>>> # Tokens present if include_syntax=True
>>> for token in annotations.tokens:
...     msg = '%11s: %s' % (token.part_of_speech, token.text_content)
...     print(msg)
DETERMINER: The
NOUN: cow
VERB: jumped
ADPOSITION: over
DETERMINER: the
NOUN: Moon
PUNCTUATION: .
>>> # Sentiment present if include_sentiment=True
>>> print(annotations.sentiment.polarity)
1
>>> print(annotations.sentiment.magnitude)
0.1
>>> # Entities present if include_entities=True
>>> for entity in annotations.entities:
...     print('=' * 20)
...     print('         name: %s' % (entity.name,))
...     print('         type: %s' % (entity.entity_type,))
...     print('wikipedia_url: %s' % (entity.wikipedia_url,))
...     print('         metadata: %s' % (entity.metadata,))
...     print('         salience: %s' % (entity.salience,))
=====
         name: Moon
         type: LOCATION
wikipedia_url: http://en.wikipedia.org/wiki/Natural_satellite
         metadata: {}
         salience: 0.11793101
```

Natural Language Client

Basic client for Google Cloud Natural Language API.

class `google.cloud.language.client.Client` (*credentials=None, http=None*)

Bases: `google.cloud.client.Client`

Client to bundle configuration needed for API requests.

Parameters

- **credentials** (`OAuth2Credentials`) – (Optional) 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.

document_from_html (*content, **kwargs*)

Create an HTML document bound to this client.

Parameters

- **content** (*str*) – The document HTML text content.
- **kwargs** (*dict*) – Remaining keyword arguments to be passed along to the `Document` constructor.

Return type `Document`

Returns An HTML document bound to this client.

Raises `TypeError` if `doc_type` is passed as a keyword argument.

document_from_text (*content, **kwargs*)

Create a plain text document bound to this client.

Parameters

- **content** (*str*) – The document plain text content.
- **kwargs** (*dict*) – Remaining keyword arguments to be passed along to the `Document` constructor.

Return type `Document`

Returns A plain-text document bound to this client.

Raises `TypeError` if `doc_type` is passed as a keyword argument.

document_from_url (*gcs_url*, *doc_type*=*'PLAIN_TEXT'*, ***kwargs*)
Create a Cloud Storage document bound to this client.

Parameters

- **gcs_url** (*str*) – The URL of the Google Cloud Storage object holding the content. Of the form `gs://{bucket}/{blob-name}`.
- **doc_type** (*str*) – (Optional) The type of text in the document. Defaults to plain text. Can also be specified as HTML via *HTML*.
- **kwargs** (*dict*) – Remaining keyword arguments to be passed along to the *Document* constructor.

Return type *Document*

Returns A document bound to this client.

87.1 Connection

Basic connection for Google Cloud Natural Language API.

class `google.cloud.language.connection.Connection` (*credentials=None*, *http=None*)
Bases: `google.cloud.connection.JSONConnection`

A connection to Google Cloud Natural Language JSON REST API.

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

API_URL_TEMPLATE = `'{api_base_url}/{api_version}/documents:{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 an API consumer.

Document

Definition for Google Cloud Natural Language API documents.

A document is used to hold text to be analyzed and annotated.

```
class google.cloud.language.document.Annotations (sentences, tokens, sentiment, entities)
```

Bases: `tuple`

Annotations for a document.

Parameters

- **sentences** (*list*) – List of *Sentence* in a document.
- **tokens** (*list*) – List of *Token* from a document.
- **sentiment** (*Sentiment*) – The sentiment of a document.
- **entities** (*list*) – List of *Entity* found in a document.

entities

Alias for field number 3

sentences

Alias for field number 0

sentiment

Alias for field number 2

tokens

Alias for field number 1

```
google.cloud.language.document.DEFAULT_LANGUAGE = 'en-US'
```

Default document language, English.

```
class google.cloud.language.document.Document (client, content=None, gcs_url=None,  

doc_type='PLAIN_TEXT', language='en-  

US', encoding='UTF8')
```

Bases: `object`

Document to send to Google Cloud Natural Language API.

Represents either plain text or HTML, and the content is either stored on the document or referred to in a Google Cloud Storage object.

Parameters

- **client** (*Client*) – A client which holds credentials and other configuration.
- **content** (*str*) – (Optional) The document text content (either plain text or HTML).

- **gcs_url** (*str*) – (Optional) The URL of the Google Cloud Storage object holding the content. Of the form `gs://{bucket}/{blob-name}`.
- **doc_type** (*str*) – (Optional) The type of text in the document. Defaults to plain text. Can be one of `PLAIN_TEXT` or `HTML`.
- **language** (*str*) – (Optional) The language of the document text. Defaults to `DEFAULT_LANGUAGE`.
- **encoding** (*str*) – (Optional) The encoding of the document text. Defaults to UTF-8. Can be one of `UTF8`, `UTF16` or `UTF32`.

Raises `ValueError` both `content` and `gcs_url` are specified or if neither are specified.

HTML = 'HTML'

HTML document type.

PLAIN_TEXT = 'PLAIN_TEXT'

Plain text document type.

TYPE_UNSPECIFIED = 'TYPE_UNSPECIFIED'

Unspecified document type.

analyze_entities ()

Analyze the entities in the current document.

Finds named entities (currently finds proper names as of August 2016) in the text, entity types, salience, mentions for each entity, and other properties.

See [analyzeEntities](#).

Return type `list`

Returns A list of `Entity` returned from the API.

analyze_sentiment ()

Analyze the sentiment in the current document.

See [analyzeSentiment](#).

Return type `Sentiment`

Returns The sentiment of the current document.

annotate_text (*include_syntax=True, include_entities=True, include_sentiment=True*)

Advanced natural language API: document syntax and other features.

Includes the full functionality of `analyze_entities()` and `analyze_sentiment()`, enabled by the flags `include_entities` and `include_sentiment` respectively.

In addition `include_syntax` adds a new feature that analyzes the document for semantic and syntactic information.

Note: This API is intended for users who are familiar with machine learning and need in-depth text features to build upon.

See [annotateText](#).

Parameters

- **include_syntax** (*bool*) – (Optional) Flag to enable syntax analysis of the current document.

- **include_entities** (*bool*) – (Optional) Flag to enable entity extraction from the current document.
- **include_sentiment** (*bool*) – (Optional) Flag to enable sentiment analysis of the current document.

Return type *Annotations*

Returns A tuple of each of the four values returned from the API: sentences, tokens, sentiment and entities.

class `google.cloud.language.document.Encoding`

Bases: `object`

Document text encoding types.

NONE = 'NONE'

Unspecified encoding type.

UTF16 = 'UTF16'

UTF-16 encoding type.

UTF32 = 'UTF32'

UTF-32 encoding type.

UTF8 = 'UTF8'

UTF-8 encoding type.

Natural Language Response Classes

89.1 Entity

Definition for Google Cloud Natural Language API entities.

An entity is used to describe a proper name extracted from text.

class `google.cloud.language.entity.Entity` (*name, entity_type, metadata, salience, mentions*)
 Bases: `object`

A Google Cloud Natural Language API entity.

Represents a phrase in text that is a known entity, such as a person, an organization, or location. The API associates information, such as salience and mentions, with entities.

The only supported metadata (as of August 2016) is `wikipedia_url`, so this value will be removed from the passed in `metadata` and put in its own property.

See [Entity message](#).

Parameters

- **name** (*str*) – The name / phrase identified as the entity.
- **entity_type** (*str*) – The type of the entity. See [EntityType enum](#).
- **metadata** (*dict*) – The metadata associated with the entity.
- **salience** (*float*) – The prominence of the entity / phrase within the text containing it.
- **mentions** (*list*) – List of strings that mention the entity.

classmethod `from_api_repr` (*payload*)

Convert an Entity from the JSON API into an `Entity`.

Parameters `payload` (*The value from the backend.*) – dict

Return type `Entity`

Returns The entity parsed from the API representation.

class `google.cloud.language.entity.EntityType`
 Bases: `object`

List of possible entity types.

CONSUMER_GOOD = 'CONSUMER_GOOD'

Consumer good entity type.

EVENT = 'EVENT'
Event entity type.

LOCATION = 'LOCATION'
Location entity type.

ORGANIZATION = 'ORGANIZATION'
Organization entity type.

OTHER = 'OTHER'
Other entity type (i.e. known but not classified).

PERSON = 'PERSON'
Person entity type.

UNKNOWN = 'UNKNOWN'
Unknown entity type.

WORK_OF_ART = 'WORK_OF_ART'
Work of art entity type.

89.2 Sentiment

Definition for Google Cloud Natural Language API sentiment.

Sentiment is the response to an `analyzeSentiment` request.

class `google.cloud.language.sentiment.Sentiment` (*polarity, magnitude*)

Bases: `object`

A Google Cloud Natural Language API sentiment object.

See [Sentiment message](#) and [Sentiment basics](#).

Parameters

- **polarity** (*float*) – Polarity of the sentiment in the `[-1.0, 1.0]` range. Larger numbers represent more positive sentiments.
- **magnitude** (*float*) – A non-negative number in the `[0, +inf)` range, which represents the absolute magnitude of sentiment regardless of polarity (positive or negative).

classmethod `from_api_repr` (*payload*)

Convert a Sentiment from the JSON API into a `Sentiment`.

Parameters `payload` (*The value from the backend.*) – dict

Return type `Sentiment`

Returns The sentiment parsed from the API representation.

89.3 Syntax

Google Cloud Natural Language API helpers for tokenized text.

The `annotateText` method, when used with the “syntax” feature, breaks a document down into tokens and sentences.

class `google.cloud.language.syntax.PartOfSpeech`

Bases: `object`

Part of speech of a `Token`.

ADJECTIVE = 'ADJ'

Part of speech: Adjective.

ADPOSITION = 'ADP'

Adposition (preposition and postposition).

ADVERB = 'ADV'

Adverb.

AFFIX = 'AFFIX'

Affix.

CARDINAL_NUMBER = 'NUM'

Cardinal number.

CONJUNCTION = 'CONJ'

Conjunction.

DETERMINER = 'DET'

Determiner.

NOUN = 'NOUN'

Noun (common and proper).

OTHER = 'X'

Other: foreign words, typos, abbreviations.

PARTICIPLE = 'PRT'

Particle or other function word.

PRONOUN = 'PRON'

Pronoun.

PUNCTUATION = 'PUNCT'

Punctuation.

UNKNOWN = 'UNKNOWN'

Unknown part of speech.

VERB = 'VERB'

Verb (all tenses and modes).

classmethod `reverse` (*tag*)

Reverses the API's enum name for the one on this class.

For example:

```
>>> PartOfSpeech.OTHER
'X'
>>> PartOfSpeech.reverse('X')
'OTHER'
```

Return type `str`

Returns The attribute name corresponding to the API part of speech enum.

class `google.cloud.language.syntax.Sentence` (*content*, *begin*)

Bases: `object`

A Google Cloud Natural Language API sentence object.

See [Sentence message](#).

Parameters

- **content** (*str*) – The text that the sentence is composed of.
- **begin** (*int*) – The beginning offset of the sentence in the original document according to the encoding type specified in the API request.

classmethod `from_api_repr` (*payload*)

Convert a sentence from the JSON API into a `Sentence`.

Parameters `payload` (*The value from the backend.*) – dict

Return type `Sentence`

Returns The sentence parsed from the API representation.

class `google.cloud.language.syntax.Token` (*text_content*, *text_begin*, *part_of_speech*, *edge_index*, *edge_label*, *lemma*)

Bases: `object`

A Google Cloud Natural Language API token object.

See [Token message](#).

Parameters

- **text_content** (*str*) – The text that the token is composed of.
- **text_begin** (*int*) – The beginning offset of the content in the original document according to the encoding type specified in the API request.
- **part_of_speech** (*str*) – The part of speech of the token. See [PartOfSpeech](#) for possible values.
- **edge_index** (*int*) – The head of this token in the dependency tree. This is the index of the token which has an arc going to this token. The index is the position of the token in the array of tokens returned by the API method. If this token is a root token, then the `edge_index` is its own index.
- **edge_label** (*str*) – See [Label enum](#).
- **lemma** (*str*) – The [Lemma](#) of the token.

classmethod `from_api_repr` (*payload*)

Convert a token from the JSON API into a `Token`.

Parameters `payload` (*The value from the backend.*) – dict

Return type `Token`

Returns The token parsed from the API representation.

Getting started

The `google-cloud` library is pip install-able:

```
$ pip install google-cloud
```

If you want to install `google-cloud-python` from source, you can clone the repository from GitHub:

```
$ git clone git://github.com/GoogleCloudPlatform/google-cloud-python.git
$ cd google-cloud-python
$ python setup.py install
```

90.1 Cloud Datastore

Google Cloud Datastore is a fully managed, schemaless database for storing non-relational data.

```
from google.cloud import datastore

client = datastore.Client()
key = client.key('Person')

entity = datastore.Entity(key=key)
entity['name'] = 'Your name'
entity['age'] = 25
client.put(entity)
```

90.2 Cloud Storage

Google Cloud Storage allows you to store data on Google infrastructure.

```
from google.cloud 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!')
```


[google.cloud.vision.face](#), 287
[google.cloud.vision.feature](#), 285
[google.cloud.vision.geometry](#), 294
[google.cloud.vision.image](#), 293
[google.cloud.vision.likelihood](#), 296
[google.cloud.vision.safe](#), 299

A

- access_grants (google.cloud.bigquery.dataset.Dataset attribute), 124
- AccessGrant (class in google.cloud.bigquery.dataset), 123
- acknowledge() (google.cloud.pubsub.subscription.Subscription method), 99
- ACL (class in google.cloud.storage.acl), 78
- acl (google.cloud.storage.blob.Blob attribute), 61
- acl (google.cloud.storage.bucket.Bucket attribute), 69
- add_entity() (google.cloud.storage.acl.ACL method), 78
- add_filter() (google.cloud.datastore.query.Query method), 44
- add_record_set() (google.cloud.dns.changes.Changes method), 199
- additions (google.cloud.dns.changes.Changes attribute), 199
- ADJECTIVE (google.cloud.language.syntax.PartOfSpeech attribute), 313
- ADPOSITION (google.cloud.language.syntax.PartOfSpeech attribute), 313
- adult (google.cloud.vision.safe.SafeSearchAnnotation attribute), 299
- ADVERB (google.cloud.language.syntax.PartOfSpeech attribute), 313
- AFFIX (google.cloud.language.syntax.PartOfSpeech attribute), 313
- align() (google.cloud.monitoring.query.Query method), 257
- Aligner (class in google.cloud.monitoring.query), 257
- all() (google.cloud.storage.acl.ACL method), 78
- all_authenticated() (google.cloud.storage.acl.ACL method), 78
- all_users() (google.cloud.pubsub.iam.Policy static method), 107
- allocate_ids() (google.cloud.datastore.client.Client method), 29
- allocate_ids() (google.cloud.datastore.connection.Connection method), 32
- allow_jagged_rows (google.cloud.bigquery.job.LoadTableFromStorageJob attribute), 129
- allow_large_results (google.cloud.bigquery.job.QueryJob attribute), 130
- allow_quoted_newlines (google.cloud.bigquery.job.LoadTableFromStorageJob attribute), 129
- alpha (google.cloud.vision.color.Color attribute), 279
- analyze_entities() (google.cloud.language.document.Document method), 308
- analyze_sentiment() (google.cloud.language.document.Document method), 308
- ancestor (google.cloud.datastore.query.Query attribute), 44
- anger_likelihood (google.cloud.vision.face.Emotions attribute), 287
- Angles (class in google.cloud.vision.face), 287
- angles (google.cloud.vision.face.Face attribute), 288
- annotate() (google.cloud.vision.client.Client method), 277
- annotate_text() (google.cloud.language.document.Document method), 308
- Annotations (class in google.cloud.language.document), 307
- API_BASE_URL (google.cloud.bigquery.connection.Connection attribute), 122
- API_BASE_URL (google.cloud.connection.JSONConnection attribute), 10
- API_BASE_URL (google.cloud.datastore.connection.Connection attribute), 32
- API_BASE_URL (google.cloud.dns.connection.Connection attribute), 192
- API_BASE_URL (google.cloud.language.connection.Connection attribute), 306
- API_BASE_URL (google.cloud.logging.connection.Connection attribute), 213
- API_BASE_URL (google.cloud.monitoring.connection.Connection attribute), 247
- API_BASE_URL (google.cloud.pubsub.connection.Connection attribute), 89
- API_BASE_URL (google.cloud.resource_manager.connection.Connection attribute), 181
- API_BASE_URL (google.cloud.storage.connection.Connection attribute), 59

API_BASE_URL (google.cloud.translate.connection.Connection attribute), 278
 attribute), 270 as_dataframe() (google.cloud.monitoring.query.Query method), 258
 API_BASE_URL (google.cloud.vision.connection.Connection attribute), 278
 attribute), 278 as_dict() (google.cloud.vision.client.VisionRequest method), 278
 API_BASE_URL (in module google.cloud.connection), 9
 api_request() (google.cloud.connection.JSONConnection method), 10
 as_dict() (google.cloud.vision.feature.Feature method), 285
 API_URL_TEMPLATE (google.cloud.bigquery.connection.Connection attribute), 122
 as_dict() (google.cloud.vision.image.Image method), 293
 attributes (google.cloud.pubsub.message.Message attribute), 103
 API_URL_TEMPLATE (google.cloud.connection.JSONConnection attribute), 10
 authenticated_users() (google.cloud.pubsub.iam.Policy static method), 107
 API_URL_TEMPLATE (google.cloud.datastore.connection.Connection attribute), 32
 auto_ack() (google.cloud.pubsub.subscription.Subscription method), 98
 API_URL_TEMPLATE (google.cloud.dns.connection.Connection attribute), 192
 AutoAck (class in google.cloud.pubsub.subscription), 101
 API_URL_TEMPLATE (google.cloud.language.connection.Connection attribute), 306
B
 API_URL_TEMPLATE (google.cloud.logging.connection.Connection attribute), 213
 BackgroundThreadTransport (class in google.cloud.logging.handlers.transports.background_thread), 229
 API_URL_TEMPLATE (google.cloud.monitoring.connection.Connection attribute), 247
 BadGateway, 13
 BadRequest, 13
 API_URL_TEMPLATE (google.cloud.pubsub.connection.Connection attribute), 89
 BadConnection (google.cloud.datastore.batch), 51
 API_URL_TEMPLATE (google.cloud.resource_manager.connection.Connection attribute), 181
 Batch (class in google.cloud.logging.logger), 215
 API_URL_TEMPLATE (google.cloud.storage.connection.Connection attribute), 59
 Batch (class in google.cloud.pubsub.topic), 95
 Batch (class in google.cloud.storage.batch), 81
 API_URL_TEMPLATE (google.cloud.translate.connection.Connection attribute), 270
 batch() (google.cloud.datastore.client.Client method), 29
 batch() (google.cloud.logging.logger.Logger method), 216
 batch() (google.cloud.pubsub.topic.Topic method), 93
 API_VERSION (google.cloud.bigquery.connection.Connection attribute), 122
 batch() (google.cloud.storage.client.Client method), 57
 begin() (google.cloud.datastore.batch.Batch method), 51
 API_VERSION (google.cloud.connection.JSONConnection attribute), 10
 begin() (google.cloud.datastore.transaction.Transaction method), 48
 API_VERSION (google.cloud.datastore.connection.Connection attribute), 32
 begin_transaction() (google.cloud.datastore.connection.Connection method), 32
 API_VERSION (google.cloud.dns.connection.Connection attribute), 192
 BIGTABLE_EMULATOR (in module google.cloud.environment_vars), 17
 API_VERSION (google.cloud.language.connection.Connection attribute), 306
 Blob (class in google.cloud.storage.blob), 61
 blob() (google.cloud.storage.bucket.Bucket method), 69
 API_VERSION (google.cloud.logging.connection.Connection attribute), 213
 blob (google.cloud.vision.color.Color attribute), 279
 blurred_likelihood (google.cloud.vision.face.FaceImageProperties attribute), 289
 API_VERSION (google.cloud.monitoring.connection.Connection attribute), 247
 Bounds (class in google.cloud.vision.face), 287
 API_VERSION (google.cloud.pubsub.connection.Connection attribute), 89
 Bounds (class in google.cloud.vision.geometry), 294
 bounds (google.cloud.vision.entity.EntityAnnotation attribute), 283
 API_VERSION (google.cloud.resource_manager.connection.Connection attribute), 181
 bounds (google.cloud.vision.face.Face attribute), 288
 API_VERSION (google.cloud.storage.connection.Connection attribute), 59
 BoundsBase (class in google.cloud.vision.geometry), 294
 Bucket (class in google.cloud.storage.bucket), 69
 API_VERSION (google.cloud.translate.connection.Connection attribute), 271
 bucket() (google.cloud.storage.client.Client method), 57
 BucketACL (class in google.cloud.storage.acl), 80
 API_VERSION (google.cloud.vision.connection.Connection

- [build_api_url\(\) \(google.cloud.connection.JSONConnection class method\)](#), 11
[build_api_url\(\) \(google.cloud.datastore.connection.Connection method\)](#), 33
[build_api_url\(\) \(google.cloud.pubsub.connection.Connection method\)](#), 89
- C**
- [cache_control \(google.cloud.storage.blob.Blob attribute\)](#), 61
[cache_hit \(google.cloud.bigquery.query.QueryResults attribute\)](#), 139
[CARDINAL_NUMBER \(google.cloud.language.syntax.PartOfSpeech attribute\)](#), 313
[Changes \(class in google.cloud.dns.changes\)](#), 199
[changes\(\) \(google.cloud.dns.zone.ManagedZone method\)](#), 193
[check_iam_permissions\(\) \(google.cloud.pubsub.subscription.Subscription method\)](#), 101
[check_iam_permissions\(\) \(google.cloud.pubsub.topic.Topic method\)](#), 95
[CHIN_GNATHION \(google.cloud.vision.face.FaceLandmarkTypes attribute\)](#), 290
[CHIN_LEFT_GONION \(google.cloud.vision.face.FaceLandmarkTypes attribute\)](#), 290
[CHIN_RIGHT_GONION \(google.cloud.vision.face.FaceLandmarkTypes attribute\)](#), 290
[chunk_size \(google.cloud.storage.blob.Blob attribute\)](#), 61
[clear\(\) \(google.cloud.storage.acl.ACL method\)](#), 78
[Client \(class in google.cloud.bigquery.client\)](#), 119
[Client \(class in google.cloud.client\)](#), 1
[Client \(class in google.cloud.datastore.client\)](#), 29
[Client \(class in google.cloud.dns.client\)](#), 191
[Client \(class in google.cloud.error_reporting.client\)](#), 235
[Client \(class in google.cloud.language.client\)](#), 305
[Client \(class in google.cloud.logging.client\)](#), 211
[Client \(class in google.cloud.monitoring.client\)](#), 241
[Client \(class in google.cloud.pubsub.client\)](#), 87
[Client \(class in google.cloud.resource_manager.client\)](#), 179
[Client \(class in google.cloud.storage.client\)](#), 57
[Client \(class in google.cloud.translate.client\)](#), 269
[Client \(class in google.cloud.vision.client\)](#), 277
[client \(google.cloud.logging.logger.Logger attribute\)](#), 216
[client \(google.cloud.logging.metric.Metric attribute\)](#), 221
[client \(google.cloud.logging.sink.Sink attribute\)](#), 223
[client \(google.cloud.storage.acl.ACL attribute\)](#), 78
[client \(google.cloud.storage.acl.BucketACL attribute\)](#), 80
[client \(google.cloud.storage.acl.ObjectACL attribute\)](#), 80
[client \(google.cloud.storage.blob.Blob attribute\)](#), 61
[client \(google.cloud.storage.bucket.Bucket attribute\)](#), 69
[ClientError, 13](#)
[CloudLoggingHandler \(class in google.cloud.logging.handlers.handlers\)](#), 225
[code \(google.cloud.exceptions.GoogleCloudError attribute\)](#), 13
[Color \(class in google.cloud.vision.color\)](#), 279
[color \(google.cloud.vision.color.ColorInformation attribute\)](#), 280
[ColorInformation \(class in google.cloud.vision.color\)](#), 280
[colors \(google.cloud.vision.color.ImagePropertiesAnnotation attribute\)](#), 280
[commit\(\) \(google.cloud.datastore.batch.Batch method\)](#), 52
[commit\(\) \(google.cloud.datastore.connection.Connection method\)](#), 33
[commit\(\) \(google.cloud.datastore.transaction.Transaction method\)](#), 48
[commit\(\) \(google.cloud.logging.logger.Batch method\)](#), 215
[commit\(\) \(google.cloud.pubsub.topic.Batch method\)](#), 95
[complete \(google.cloud.bigquery.query.QueryResults attribute\)](#), 139
[complete \(google.cloud.operation.Operation attribute\)](#), 27
[complete_key\(\) \(google.cloud.datastore.key.Key method\)](#), 39
[component_count \(google.cloud.storage.blob.Blob attribute\)](#), 61
[Compression \(class in google.cloud.bigquery.job\)](#), 127
[compression \(google.cloud.bigquery.job.ExtractTableToStorageJob attribute\)](#), 128
[configure_website\(\) \(google.cloud.storage.bucket.Bucket method\)](#), 69
[Conflict, 13](#)
[CONJUNCTION \(google.cloud.language.syntax.PartOfSpeech attribute\)](#), 313
[Connection \(class in google.cloud.bigquery.connection\)](#), 122
[Connection \(class in google.cloud.connection\)](#), 9
[Connection \(class in google.cloud.datastore.connection\)](#), 32
[Connection \(class in google.cloud.dns.connection\)](#), 192
[Connection \(class in google.cloud.language.connection\)](#), 306
[Connection \(class in google.cloud.logging.connection\)](#), 213
[Connection \(class in google.cloud.monitoring.connection\)](#), 246
[Connection \(class in google.cloud.pubsub.connection\)](#), 88
[Connection \(class in google.cloud.resource_manager.connection\)](#), 180
[Connection \(class in google.cloud.storage.connection\)](#), 59

- Connection (class in google.cloud.translate.connection), 270
 - Connection (class in google.cloud.vision.connection), 278
 - connection (google.cloud.datastore.batch.Batch attribute), 52
 - connection (google.cloud.datastore.transaction.Transaction attribute), 48
 - connection (google.cloud.storage.client.Client attribute), 57
 - CONSUMER_GOOD (google.cloud.language.entity.EntityType attribute), 311
 - content (google.cloud.vision.image.Image attribute), 293
 - content_disposition (google.cloud.storage.blob.Blob attribute), 61
 - content_encoding (google.cloud.storage.blob.Blob attribute), 62
 - content_language (google.cloud.storage.blob.Blob attribute), 62
 - content_type (google.cloud.storage.blob.Blob attribute), 62
 - copy() (google.cloud.monitoring.query.Query method), 259
 - copy_blob() (google.cloud.storage.bucket.Bucket method), 70
 - copy_table() (google.cloud.bigquery.client.Client method), 119
 - CopyJob (class in google.cloud.bigquery.job), 127
 - cors (google.cloud.storage.bucket.Bucket attribute), 70
 - crc32c (google.cloud.storage.blob.Blob attribute), 62
 - create() (google.cloud.bigquery.dataset.Dataset method), 124
 - create() (google.cloud.bigquery.table.Table method), 133
 - create() (google.cloud.dns.changes.Changes method), 199
 - create() (google.cloud.dns.zone.ManagedZone method), 193
 - create() (google.cloud.logging.metric.Metric method), 221
 - create() (google.cloud.logging.sink.Sink method), 223
 - create() (google.cloud.monitoring.group.Group method), 253
 - create() (google.cloud.monitoring.metric.MetricDescriptor method), 250
 - create() (google.cloud.pubsub.subscription.Subscription method), 98
 - create() (google.cloud.pubsub.topic.Topic method), 92
 - create() (google.cloud.resource_manager.project.Project method), 183
 - create() (google.cloud.storage.bucket.Bucket method), 70
 - create_bucket() (google.cloud.storage.client.Client method), 58
 - create_disposition (google.cloud.bigquery.job.CopyJob attribute), 127
 - create_disposition (google.cloud.bigquery.job.LoadTableFromStorageJob attribute), 129
 - create_disposition (google.cloud.bigquery.job.QueryJob attribute), 130
 - created (google.cloud.bigquery.dataset.Dataset attribute), 124
 - created (google.cloud.bigquery.table.Table attribute), 133
 - created (google.cloud.dns.zone.ManagedZone attribute), 193
 - CreateDisposition (class in google.cloud.bigquery.job), 127
 - credentials (google.cloud.connection.Connection attribute), 9
 - CREDENTIALS (in module google.cloud.environment_vars), 17
 - current() (google.cloud.datastore.batch.Batch method), 52
 - current() (google.cloud.datastore.transaction.Transaction method), 48
 - current() (google.cloud.storage.batch.Batch method), 81
 - current_batch (google.cloud.datastore.client.Client attribute), 29
 - current_batch (google.cloud.storage.client.Client attribute), 58
 - current_transaction (google.cloud.datastore.client.Client attribute), 29
- ## D
- Dataset (class in google.cloud.bigquery.dataset), 123
 - dataset() (google.cloud.bigquery.client.Client method), 119
 - dataset_id (google.cloud.bigquery.dataset.Dataset attribute), 124
 - dataset_name (google.cloud.bigquery.table.Table attribute), 133
 - DATASTORE_API_HOST (in module google.cloud.datastore.connection), 35
 - default_dataset (google.cloud.bigquery.job.QueryJob attribute), 130
 - default_dataset (google.cloud.bigquery.query.QueryResults attribute), 139
 - DEFAULT_LANGUAGE (in module google.cloud.language.document), 307
 - default_object_acl (google.cloud.storage.bucket.Bucket attribute), 70
 - default_table_expiration_ms (google.cloud.bigquery.dataset.Dataset attribute), 124
 - DEFAULT_USER_AGENT (in module google.cloud.connection), 10
 - DefaultObjectACL (class in google.cloud.storage.acl), 80
 - delete() (google.cloud.bigquery.dataset.Dataset method), 124
 - delete() (google.cloud.bigquery.table.Table method), 133

- delete() (google.cloud.datastore.batch.Batch method), 52
- delete() (google.cloud.datastore.client.Client method), 30
- delete() (google.cloud.datastore.transaction.Transaction method), 49
- delete() (google.cloud.dns.zone.ManagedZone method), 193
- delete() (google.cloud.logging.logger.Logger method), 216
- delete() (google.cloud.logging.metric.Metric method), 221
- delete() (google.cloud.logging.sink.Sink method), 223
- delete() (google.cloud.monitoring.group.Group method), 253
- delete() (google.cloud.monitoring.metric.MetricDescriptor method), 250
- delete() (google.cloud.pubsub.subscription.Subscription method), 98
- delete() (google.cloud.pubsub.topic.Topic method), 92
- delete() (google.cloud.resource_manager.project.Project method), 183
- delete() (google.cloud.storage.blob.Blob method), 62
- delete() (google.cloud.storage.bucket.Bucket method), 70
- delete_blob() (google.cloud.storage.bucket.Bucket method), 71
- delete_blobs() (google.cloud.storage.bucket.Bucket method), 71
- delete_multi() (google.cloud.datastore.client.Client method), 30
- delete_record_set() (google.cloud.dns.changes.Changes method), 199
- deletions (google.cloud.dns.changes.Changes attribute), 199
- description (google.cloud.bigquery.dataset.Dataset attribute), 124
- description (google.cloud.bigquery.table.Table attribute), 133
- description (google.cloud.dns.zone.ManagedZone attribute), 194
- description (google.cloud.vision.entity.EntityAnnotation attribute), 283
- destination (google.cloud.bigquery.job.QueryJob attribute), 130
- destination_format (google.cloud.bigquery.job.ExtractTableToStorageJob attribute), 128
- DestinationFormat (class in google.cloud.bigquery.job), 127
- detect_faces() (google.cloud.vision.image.Image method), 293
- detect_labels() (google.cloud.vision.image.Image method), 293
- detect_landmarks() (google.cloud.vision.image.Image method), 293
- detect_language() (google.cloud.translate.client.Client method), 269
- detect_logos() (google.cloud.vision.image.Image method), 294
- detect_properties() (google.cloud.vision.image.Image method), 294
- detect_safe_search() (google.cloud.vision.image.Image method), 294
- detect_text() (google.cloud.vision.image.Image method), 294
- detection_confidence (google.cloud.vision.face.Face attribute), 288
- DETERMINER (google.cloud.language.syntax.PartOfSpeech attribute), 313
- DISABLE_GRPC (in module google.cloud.environment_vars), 17
- disable_logging() (google.cloud.storage.bucket.Bucket method), 72
- disable_website() (google.cloud.storage.bucket.Bucket method), 72
- distinct_on (google.cloud.datastore.query.Query attribute), 44
- Document (class in google.cloud.language.document), 307
- document_from_html() (google.cloud.language.client.Client method), 305
- document_from_text() (google.cloud.language.client.Client method), 305
- document_from_url() (google.cloud.language.client.Client method), 305
- domain() (google.cloud.pubsub.iam.Policy static method), 106
- domain() (google.cloud.storage.acl.ACL method), 78
- download_as_string() (google.cloud.storage.blob.Blob method), 62
- download_to_file() (google.cloud.storage.blob.Blob method), 63
- download_to_filename() (google.cloud.storage.blob.Blob method), 63
- dry_run (google.cloud.bigquery.query.QueryResults attribute), 139
- ## E
- EDITOR_ROLE (in module google.cloud.pubsub.iam), 105
- emit() (google.cloud.logging.handlers.handlers.CloudLoggingHandler method), 225
- Emotions (class in google.cloud.vision.face), 287
- emotions (google.cloud.vision.face.Face attribute), 289
- enable_logging() (google.cloud.storage.bucket.Bucket method), 72
- Encoding (class in google.cloud.bigquery.job), 128
- Encoding (class in google.cloud.language.document), 309
- encoding (google.cloud.bigquery.job.LoadTableFromStorageJob attribute), 129

- ENGLISH_ISO_639 (in module google.cloud.translate.client), 270
 - entities (google.cloud.language.document.Annotations attribute), 307
 - Entity (class in google.cloud.datastore.entity), 37
 - Entity (class in google.cloud.language.entity), 311
 - entity() (google.cloud.storage.acl.ACL method), 78
 - entity_from_dict() (google.cloud.storage.acl.ACL method), 79
 - entity_from_protobuf() (in module google.cloud.datastore.helpers), 55
 - ENTITY_TYPES (google.cloud.bigquery.dataset.AccessGrant attribute), 123
 - EntityAnnotation (class in google.cloud.vision.entity), 283
 - EntityType (class in google.cloud.language.entity), 311
 - environment variable
 - GOOGLE_APPLICATION_CREDENTIALS, 6, 109
 - GOOGLE_CLOUD_DISABLE_GRPC, 83, 201
 - GOOGLE_CLOUD_PROJECT, 83, 109, 147, 201, 233, 238, 273
 - errors (google.cloud.bigquery.query.QueryResults attribute), 139
 - errors (google.cloud.exceptions.GoogleCloudError attribute), 13
 - etag (google.cloud.bigquery.dataset.Dataset attribute), 124
 - etag (google.cloud.bigquery.table.Table attribute), 134
 - etag (google.cloud.storage.blob.Blob attribute), 63
 - etag (google.cloud.storage.bucket.Bucket attribute), 72
 - EVENT (google.cloud.language.entity.EntityType attribute), 311
 - exclude_from_indexes (google.cloud.datastore.entity.Entity attribute), 38
 - exists() (google.cloud.bigquery.dataset.Dataset method), 124
 - exists() (google.cloud.bigquery.table.Table method), 134
 - exists() (google.cloud.dns.changes.Changes method), 200
 - exists() (google.cloud.dns.zone.ManagedZone method), 194
 - exists() (google.cloud.logging.metric.Metric method), 221
 - exists() (google.cloud.logging.sink.Sink method), 223
 - exists() (google.cloud.monitoring.group.Group method), 254
 - exists() (google.cloud.pubsub.subscription.Subscription method), 98
 - exists() (google.cloud.pubsub.topic.Topic method), 92
 - exists() (google.cloud.resource_manager.project.Project method), 184
 - exists() (google.cloud.storage.blob.Blob method), 64
 - exists() (google.cloud.storage.bucket.Bucket method), 72
 - expires (google.cloud.bigquery.table.Table attribute), 134
 - extract_table_to_storage() (google.cloud.bigquery.client.Client method), 119
 - ExtractTableToStorageJob (class in google.cloud.bigquery.job), 128
- ## F
- Face (class in google.cloud.vision.face), 288
 - FACE_DETECTION (google.cloud.vision.feature.FeatureTypes attribute), 285
 - FaceImageProperties (class in google.cloud.vision.face), 289
 - FaceLandmarkTypes (class in google.cloud.vision.face), 290
 - fd_bounds (google.cloud.vision.face.Face attribute), 289
 - FDBounds (class in google.cloud.vision.face), 288
 - FDBounds (class in google.cloud.vision.geometry), 295
 - Feature (class in google.cloud.vision.feature), 285
 - feature_type (google.cloud.vision.feature.Feature attribute), 285
 - features (google.cloud.vision.client.VisionRequest attribute), 278
 - FeatureTypes (class in google.cloud.vision.feature), 285
 - fetch() (google.cloud.datastore.query.Query method), 45
 - fetch_data() (google.cloud.bigquery.query.QueryResults method), 139
 - fetch_data() (google.cloud.bigquery.table.Table method), 134
 - fetch_group() (google.cloud.monitoring.client.Client method), 241
 - fetch_metric_descriptor() (google.cloud.monitoring.client.Client method), 242
 - fetch_parent() (google.cloud.monitoring.group.Group method), 254
 - fetch_project() (google.cloud.resource_manager.client.Client method), 179
 - fetch_resource_descriptor() (google.cloud.monitoring.client.Client method), 242
 - field_delimiter (google.cloud.bigquery.job.ExtractTableToStorageJob attribute), 128
 - field_delimiter (google.cloud.bigquery.job.LoadTableFromStorageJob attribute), 129
 - filter (google.cloud.monitoring.query.Query attribute), 259
 - filters (google.cloud.datastore.query.Query attribute), 45
 - finish() (google.cloud.storage.batch.Batch method), 81
 - flat_path (google.cloud.datastore.key.Key attribute), 40
 - flatten_results (google.cloud.bigquery.job.QueryJob attribute), 130
 - Forbidden, 13
 - FOREHEAD_GLABELLA (google.cloud.vision.face.FaceLandmarkTypes

- attribute), 290
 - friendly_name (google.cloud.bigquery.dataset.Dataset attribute), 125
 - friendly_name (google.cloud.bigquery.table.Table attribute), 134
 - from_api_repr() (google.cloud.bigquery.client.Project class method), 122
 - from_api_repr() (google.cloud.bigquery.dataset.Dataset class method), 125
 - from_api_repr() (google.cloud.bigquery.job.CopyJob class method), 127
 - from_api_repr() (google.cloud.bigquery.job.ExtractTableToStorageJob class method), 128
 - from_api_repr() (google.cloud.bigquery.job.LoadTableFromStorageJob class method), 129
 - from_api_repr() (google.cloud.bigquery.job.QueryJob class method), 130
 - from_api_repr() (google.cloud.bigquery.table.Table class method), 135
 - from_api_repr() (google.cloud.dns.changes.Changes class method), 200
 - from_api_repr() (google.cloud.dns.resource_record_set.ResourceRecordSet class method), 197
 - from_api_repr() (google.cloud.dns.zone.ManagedZone class method), 194
 - from_api_repr() (google.cloud.language.entity.Entity class method), 311
 - from_api_repr() (google.cloud.language.sentiment.Sentiment class method), 312
 - from_api_repr() (google.cloud.language.syntax.Sentence class method), 314
 - from_api_repr() (google.cloud.language.syntax.Token class method), 314
 - from_api_repr() (google.cloud.logging.metric.Metric class method), 222
 - from_api_repr() (google.cloud.logging.sink.Sink class method), 224
 - from_api_repr() (google.cloud.pubsub.iam.Policy class method), 107
 - from_api_repr() (google.cloud.pubsub.message.Message class method), 103
 - from_api_repr() (google.cloud.pubsub.subscription.Subscription class method), 97
 - from_api_repr() (google.cloud.pubsub.topic.Topic class method), 92
 - from_api_repr() (google.cloud.resource_manager.project.Project class method), 184
 - from_api_repr() (google.cloud.vision.color.Color class method), 279
 - from_api_repr() (google.cloud.vision.color.ColorInformation class method), 280
 - from_api_repr() (google.cloud.vision.color.ImagePropertiesAnnotation class method), 280
 - from_api_repr() (google.cloud.vision.entity.EntityAnnotation class method), 283
 - from_api_repr() (google.cloud.vision.face.Angles class method), 287
 - from_api_repr() (google.cloud.vision.face.Emotions class method), 288
 - from_api_repr() (google.cloud.vision.face.Face class method), 289
 - from_api_repr() (google.cloud.vision.face.FaceImageProperties class method), 289
 - from_api_repr() (google.cloud.vision.face.Landmark class method), 291
 - from_api_repr() (google.cloud.vision.geometry.BoundsBase class method), 295
 - from_api_repr() (google.cloud.vision.geometry.LocationInformation class method), 295
 - from_api_repr() (google.cloud.vision.geometry.Position class method), 296
 - from_api_repr() (google.cloud.vision.safe.SafeSearchAnnotation class method), 299
 - from_pb() (google.cloud.operation.Operation class method), 27
 - from_query_job() (google.cloud.bigquery.query.QueryResults class method), 140
 - from_service_account_json() (google.cloud.client.Client method), 1
 - from_service_account_json() (google.cloud.client.JSONClient method), 2
 - from_service_account_p12() (google.cloud.client.Client method), 1
 - from_service_account_p12() (google.cloud.client.JSONClient method), 2
 - full_name (google.cloud.logging.logger.Logger attribute), 216
 - full_name (google.cloud.logging.metric.Metric attribute), 222
 - full_name (google.cloud.logging.sink.Sink attribute), 224
 - full_name (google.cloud.pubsub.subscription.Subscription attribute), 97
 - full_name (google.cloud.pubsub.topic.Topic attribute), 92
 - full_name (google.cloud.resource_manager.project.Project attribute), 184
- ## G
- GCD_DATASET (in module google.cloud.environment_vars), 17
 - GCD_HOST (in module google.cloud.environment_vars), 17
 - generate_signed_url() (google.cloud.storage.blob.Blob method), 64
 - generate_signed_url() (in module google.cloud.credentials), 5
 - generation (google.cloud.storage.blob.Blob attribute), 64

- get() (google.cloud.datastore.client.Client method), 30
- get_blob() (google.cloud.storage.bucket.Bucket method), 72
- get_bucket() (google.cloud.storage.client.Client method), 58
- get_credentials() (in module google.cloud.credentials), 6
- get_entities() (google.cloud.storage.acl.ACL method), 79
- get_entity() (google.cloud.storage.acl.ACL method), 79
- get_iam_policy() (google.cloud.pubsub.subscription.Subscription method), 100
- get_iam_policy() (google.cloud.pubsub.topic.Topic method), 94
- get_languages() (google.cloud.translate.client.Client method), 269
- get_logging() (google.cloud.storage.bucket.Bucket method), 73
- get_multi() (google.cloud.datastore.client.Client method), 30
- google.cloud.bigquery.client (module), 119
- google.cloud.bigquery.connection (module), 122
- google.cloud.bigquery.dataset (module), 123
- google.cloud.bigquery.job (module), 127
- google.cloud.bigquery.query (module), 139
- google.cloud.bigquery.schema (module), 143
- google.cloud.bigquery.table (module), 133
- google.cloud.client (module), 1
- google.cloud.connection (module), 9
- google.cloud.credentials (module), 5
- google.cloud.datastore.batch (module), 51
- google.cloud.datastore.client (module), 29
- google.cloud.datastore.connection (module), 32
- google.cloud.datastore.entity (module), 37
- google.cloud.datastore.helpers (module), 55
- google.cloud.datastore.key (module), 39
- google.cloud.datastore.query (module), 43
- google.cloud.datastore.transaction (module), 47
- google.cloud.dns.changes (module), 199
- google.cloud.dns.client (module), 191
- google.cloud.dns.connection (module), 192
- google.cloud.dns.resource_record_set (module), 197
- google.cloud.dns.zone (module), 193
- google.cloud.environment_vars (module), 17
- google.cloud.error_reporting.client (module), 235
- google.cloud.exceptions (module), 13
- google.cloud.language.client (module), 305
- google.cloud.language.connection (module), 306
- google.cloud.language.document (module), 307
- google.cloud.language.entity (module), 311
- google.cloud.language.sentiment (module), 312
- google.cloud.language.syntax (module), 312
- google.cloud.logging.client (module), 211
- google.cloud.logging.connection (module), 213
- google.cloud.logging.entries (module), 219
- google.cloud.logging.handlers.handlers (module), 225
- google.cloud.logging.handlers.transports.background_thread (module), 229
- google.cloud.logging.handlers.transports.base (module), 231
- google.cloud.logging.handlers.transports.sync (module), 227
- google.cloud.logging.logger (module), 215
- google.cloud.logging.metric (module), 221
- google.cloud.logging.sink (module), 223
- google.cloud.monitoring.client (module), 241
- google.cloud.monitoring.connection (module), 246
- google.cloud.monitoring.group (module), 253
- google.cloud.monitoring.label (module), 265
- google.cloud.monitoring.metric (module), 249
- google.cloud.monitoring.query (module), 257
- google.cloud.monitoring.resource (module), 251
- google.cloud.monitoring.timeseries (module), 263
- google.cloud.operation (module), 27
- google.cloud.pubsub.client (module), 87
- google.cloud.pubsub.connection (module), 88
- google.cloud.pubsub.iam (module), 105
- google.cloud.pubsub.message (module), 103
- google.cloud.pubsub.subscription (module), 97
- google.cloud.pubsub.topic (module), 91
- google.cloud.resource_manager.client (module), 179
- google.cloud.resource_manager.connection (module), 180
- google.cloud.resource_manager.project (module), 183
- google.cloud.storage.acl (module), 77
- google.cloud.storage.batch (module), 81
- google.cloud.storage.blob (module), 61
- google.cloud.storage.bucket (module), 69
- google.cloud.storage.client (module), 57
- google.cloud.storage.connection (module), 59
- google.cloud.translate.client (module), 269
- google.cloud.translate.connection (module), 270
- google.cloud.vision.client (module), 277
- google.cloud.vision.color (module), 279
- google.cloud.vision.connection (module), 278
- google.cloud.vision.entity (module), 283
- google.cloud.vision.face (module), 287
- google.cloud.vision.feature (module), 285
- google.cloud.vision.geometry (module), 294
- google.cloud.vision.image (module), 293
- google.cloud.vision.likelihood (module), 296
- google.cloud.vision.safe (module), 299
- GOOGLE_APPLICATION_CREDENTIALS, 6, 109
- GOOGLE_CLOUD_DISABLE_GRPC, 83, 201
- GOOGLE_CLOUD_PROJECT, 83, 109, 147, 201, 233, 238, 273
- GoogleCloudError, 13
- green (google.cloud.vision.color.Color attribute), 279
- Group (class in google.cloud.monitoring.group), 253

group() (google.cloud.monitoring.client.Client method), 242

group() (google.cloud.pubsub.iam.Policy static method), 106

group() (google.cloud.storage.acl.ACL method), 79

GrpcRendezvous (in module google.cloud.exceptions), 13

H

has_entity() (google.cloud.storage.acl.ACL method), 79

header() (google.cloud.monitoring.timeseries.TimeSeries method), 263

headwear_likelihood (google.cloud.vision.face.Face attribute), 289

HTML (google.cloud.language.document.Document attribute), 308

http (google.cloud.connection.Connection attribute), 10

HTTPContext (class in google.cloud.error_reporting.client), 236

I

iam_policy_api (google.cloud.pubsub.client.Client attribute), 87

id (google.cloud.datastore.key.Key attribute), 40

id (google.cloud.datastore.transaction.Transaction attribute), 49

id (google.cloud.monitoring.group.Group attribute), 254

id (google.cloud.storage.blob.Blob attribute), 65

id (google.cloud.storage.bucket.Bucket attribute), 73

id_or_name (google.cloud.datastore.key.Key attribute), 40

ignore_unknown_values (google.cloud.bigquery.job.LoadTableFromStorageJob attribute), 129

Image (class in google.cloud.vision.image), 293

image (google.cloud.vision.client.VisionRequest attribute), 278

image() (google.cloud.vision.client.Client method), 277

image_properties (google.cloud.vision.face.Face attribute), 289

IMAGE_PROPERTIES (google.cloud.vision.feature.FeatureTypes attribute), 285

ImagePropertiesAnnotation (class in google.cloud.vision.color), 280

input_file_bytes (google.cloud.bigquery.job.LoadTableFromStorageJob attribute), 129

input_files (google.cloud.bigquery.job.LoadTableFromStorageJob attribute), 129

insert_data() (google.cloud.bigquery.table.Table method), 135

InternalServerError, 14

is_partial (google.cloud.datastore.key.Key attribute), 40

iter() (google.cloud.monitoring.query.Query method), 259

Iterator (class in google.cloud.datastore.query), 43

J

job (google.cloud.bigquery.query.QueryResults attribute), 140

job_from_resource() (google.cloud.bigquery.client.Client method), 120

joy_likelihood (google.cloud.vision.face.Emotions attribute), 288

JSONClient (class in google.cloud.client), 2

JSONConnection (class in google.cloud.connection), 10

K

Key (class in google.cloud.datastore.key), 39

key() (google.cloud.datastore.client.Client method), 31

key_filter() (google.cloud.datastore.query.Query method), 45

key_from_protobuf() (in module google.cloud.datastore.helpers), 55

keys_only() (google.cloud.datastore.query.Query method), 45

kind (google.cloud.datastore.entity.Entity attribute), 38

kind (google.cloud.datastore.key.Key attribute), 40

kind (google.cloud.datastore.query.Query attribute), 45

L

LABEL_DETECTION (google.cloud.vision.feature.FeatureTypes attribute), 285

LabelDescriptor (class in google.cloud.monitoring.label), 265

labels (google.cloud.monitoring.timeseries.TimeSeries attribute), 264

LabelValueType (class in google.cloud.monitoring.label), 265

Landmark (class in google.cloud.vision.face), 291

LANDMARK_DETECTION (google.cloud.vision.feature.FeatureTypes attribute), 286

landmark_type (google.cloud.vision.face.Landmark attribute), 291

landmarking_confidence (google.cloud.vision.face.Face attribute), 289

Landmarks (class in google.cloud.vision.face), 291

landmarks (google.cloud.vision.face.Face attribute), 289

latitude (google.cloud.vision.geometry.LocationInformation attribute), 295

LEFT_EAR_TRAGION (google.cloud.vision.face.FaceLandmarkTypes attribute), 290

LEFT_EYE (google.cloud.vision.face.FaceLandmarkTypes attribute), 290

LEFT_EYE_BOTTOM_BOUNDARY (google.cloud.vision.face.FaceLandmarkTypes attribute), 290

LEFT_EYE_LEFT_CORNER (google.cloud.vision.face.FaceLandmarkTypes attribute), 290

- LEFT_EYE_PUPIL (google.cloud.vision.face.FaceLandmarkTypes attribute), 290
- LEFT_EYE_RIGHT_CORNER (google.cloud.vision.face.FaceLandmarkTypes attribute), 290
- LEFT_EYE_TOP_BOUNDARY (google.cloud.vision.face.FaceLandmarkTypes attribute), 290
- LEFT_EYEBROW_UPPER_MIDPOINT (google.cloud.vision.face.FaceLandmarkTypes attribute), 290
- LEFT_OF_LEFT_EYEBROW (google.cloud.vision.face.FaceLandmarkTypes attribute), 290
- LEFT_OF_RIGHT_EYEBROW (google.cloud.vision.face.FaceLandmarkTypes attribute), 290
- LengthRequired, 14
- lifecycle_rules (google.cloud.storage.bucket.Bucket attribute), 73
- Likelihood (class in google.cloud.vision.likelihood), 296
- LIKELY (google.cloud.vision.likelihood.Likelihood attribute), 297
- list_ancestors() (google.cloud.monitoring.group.Group method), 254
- list_blobs() (google.cloud.storage.bucket.Bucket method), 73
- list_buckets() (google.cloud.storage.client.Client method), 58
- list_changes() (google.cloud.dns.zone.ManagedZone method), 194
- list_children() (google.cloud.monitoring.group.Group method), 254
- list_datasets() (google.cloud.bigquery.client.Client method), 120
- list_descendants() (google.cloud.monitoring.group.Group method), 254
- list_entries() (google.cloud.logging.client.Client method), 211
- list_entries() (google.cloud.logging.logger.Logger method), 216
- list_groups() (google.cloud.monitoring.client.Client method), 242
- list_jobs() (google.cloud.bigquery.client.Client method), 120
- list_members() (google.cloud.monitoring.group.Group method), 254
- list_metric_descriptors() (google.cloud.monitoring.client.Client method), 243
- list_metrics() (google.cloud.logging.client.Client method), 211
- list_partitions() (google.cloud.bigquery.table.Table method), 135
- list_projects() (google.cloud.bigquery.client.Client method), 121
- list_projects() (google.cloud.resource_manager.client.Client method), 179
- list_resource_descriptors() (google.cloud.monitoring.client.Client method), 243
- list_resource_record_sets() (google.cloud.dns.zone.ManagedZone method), 194
- list_sinks() (google.cloud.logging.client.Client method), 212
- list_subscriptions() (google.cloud.pubsub.client.Client method), 87
- list_subscriptions() (google.cloud.pubsub.topic.Topic method), 93
- list_tables() (google.cloud.bigquery.dataset.Dataset method), 125
- list_topics() (google.cloud.pubsub.client.Client method), 88
- list_zones() (google.cloud.dns.client.Client method), 191
- load_table_from_storage() (google.cloud.bigquery.client.Client method), 121
- LoadTableFromStorageJob (class in google.cloud.bigquery.job), 128
- locale (google.cloud.vision.entity.EntityAnnotation attribute), 283
- location (google.cloud.bigquery.dataset.Dataset attribute), 125
- location (google.cloud.bigquery.table.Table attribute), 135
- LOCATION (google.cloud.language.entity.EntityType attribute), 312
- location (google.cloud.storage.bucket.Bucket attribute), 74
- LocationInformation (class in google.cloud.vision.geometry), 295
- locations (google.cloud.vision.entity.EntityAnnotation attribute), 284
- log_proto() (google.cloud.logging.logger.Batch method), 215
- log_proto() (google.cloud.logging.logger.Logger method), 217
- log_struct() (google.cloud.logging.logger.Batch method), 215
- log_struct() (google.cloud.logging.logger.Logger method), 217
- log_text() (google.cloud.logging.logger.Batch method), 216
- log_text() (google.cloud.logging.logger.Logger method), 217
- Logger (class in google.cloud.logging.logger), 216
- logger() (google.cloud.logging.client.Client method), 212
- logger_name_from_path() (in module

- google.cloud.logging.entries), 219
 - logging_api (google.cloud.logging.client.Client attribute), 212
 - LOGO_DETECTION (google.cloud.vision.feature.FeatureTypes attribute), 286
 - longitude (google.cloud.vision.geometry.LocationInformation attribute), 295
 - lookup() (google.cloud.datastore.connection.Connection method), 33
 - lookup_bucket() (google.cloud.storage.client.Client method), 59
 - LOWER_LIP (google.cloud.vision.face.FaceLandmarkTypes attribute), 290
- ## M
- make_exception() (in module google.cloud.exceptions), 15
 - make_public() (google.cloud.storage.blob.Blob method), 65
 - make_public() (google.cloud.storage.bucket.Bucket method), 74
 - ManagedZone (class in google.cloud.dns.zone), 193
 - max_bad_records (google.cloud.bigquery.job.LoadTableFromStorageJob attribute), 129
 - max_results (google.cloud.bigquery.query.QueryResults attribute), 140
 - max_results (google.cloud.vision.feature.Feature attribute), 285
 - maximum_billing_tier (google.cloud.bigquery.job.QueryJob attribute), 131
 - maximum_bytes_billed (google.cloud.bigquery.job.QueryJob attribute), 131
 - md5_hash (google.cloud.storage.blob.Blob attribute), 65
 - media_link (google.cloud.storage.blob.Blob attribute), 65
 - medical (google.cloud.vision.safe.SafeSearchAnnotation attribute), 299
 - Message (class in google.cloud.pubsub.message), 103
 - metadata (google.cloud.storage.blob.Blob attribute), 65
 - metageneration (google.cloud.storage.blob.Blob attribute), 65
 - metageneration (google.cloud.storage.bucket.Bucket attribute), 74
 - MethodNotAllowed, 14
 - MethodNotImplemented, 14
 - Metric (class in google.cloud.logging.metric), 221
 - Metric (class in google.cloud.monitoring.metric), 249
 - metric() (google.cloud.logging.client.Client method), 212
 - metric() (google.cloud.monitoring.client.Client static method), 243
 - metric_descriptor() (google.cloud.monitoring.client.Client method), 244
 - METRIC_KIND_UNSPECIFIED (google.cloud.monitoring.metric.MetricKind attribute), 250
 - metric_type (google.cloud.monitoring.query.Query attribute), 259
 - MetricDescriptor (class in google.cloud.monitoring.metric), 249
 - MetricKind (class in google.cloud.monitoring.metric), 250
 - metrics_api (google.cloud.logging.client.Client attribute), 213
 - mid (google.cloud.vision.entity.EntityAnnotation attribute), 284
 - MIDPOINT_BETWEEN_EYES (google.cloud.vision.face.FaceLandmarkTypes attribute), 290
 - MIMEApplicationHTTP (class in google.cloud.storage.batch), 81
 - modified (google.cloud.bigquery.dataset.Dataset attribute), 125
 - modified (google.cloud.bigquery.table.Table attribute), 135
 - modify_ack_deadline() (google.cloud.pubsub.subscription.Subscription method), 100
 - modify_push_configuration() (google.cloud.pubsub.subscription.Subscription method), 99
 - MOUTH_CENTER (google.cloud.vision.face.FaceLandmarkTypes attribute), 290
 - MOUTH_LEFT (google.cloud.vision.face.FaceLandmarkTypes attribute), 290
 - MOUTH_RIGHT (google.cloud.vision.face.FaceLandmarkTypes attribute), 290
 - MovedPermanently, 14
 - mutations (google.cloud.datastore.batch.Batch attribute), 52
 - mutations (google.cloud.datastore.transaction.Transaction attribute), 49
- ## N
- name (google.cloud.bigquery.query.QueryResults attribute), 140
 - name (google.cloud.datastore.key.Key attribute), 40
 - name (google.cloud.dns.changes.Changes attribute), 200
 - name (google.cloud.monitoring.group.Group attribute), 255
 - name_server_set (google.cloud.dns.zone.ManagedZone attribute), 195
 - name_servers (google.cloud.dns.zone.ManagedZone attribute), 195
 - namespace (google.cloud.datastore.batch.Batch attribute), 52
 - namespace (google.cloud.datastore.key.Key attribute), 40
 - namespace (google.cloud.datastore.query.Query attribute), 45
 - namespace (google.cloud.datastore.transaction.Transaction attribute), 49

- new_project() (google.cloud.resource_manager.client.Clientparent_name (google.cloud.monitoring.group.Group attribute), 180 method), 180
- next_page() (google.cloud.datastore.query.Iterator method), 43
- NoContent (class in google.cloud.storage.batch), 81
- NONE (google.cloud.language.document.Encoding attribute), 309
- NOSE_BOTTOM_CENTER (google.cloud.vision.face.FaceLandmarkTypes attribute), 290
- NOSE_BOTTOM_LEFT (google.cloud.vision.face.FaceLandmarkTypes attribute), 290
- NOSE_BOTTOM_RIGHT (google.cloud.vision.face.FaceLandmarkTypes attribute), 290
- NOSE_TIP (google.cloud.vision.face.FaceLandmarkTypes attribute), 290
- NotFound, 14
- NotModified, 14
- NOUN (google.cloud.language.syntax.PartOfSpeech attribute), 313
- num_bytes (google.cloud.bigquery.table.Table attribute), 136
- num_rows (google.cloud.bigquery.table.Table attribute), 136
- O**
- ObjectACL (class in google.cloud.storage.acl), 80
- Operation (class in google.cloud.operation), 27
- OPERATORS (google.cloud.datastore.query.Query attribute), 44
- order (google.cloud.datastore.query.Query attribute), 46
- ORGANIZATION (google.cloud.language.entity.EntityType attribute), 312
- OTHER (google.cloud.language.entity.EntityType attribute), 312
- OTHER (google.cloud.language.syntax.PartOfSpeech attribute), 313
- output_bytes (google.cloud.bigquery.job.LoadTableFromStorageJob attribute), 129
- output_rows (google.cloud.bigquery.job.LoadTableFromStorageJob attribute), 130
- owner (google.cloud.storage.blob.Blob attribute), 65
- owner (google.cloud.storage.bucket.Bucket attribute), 74
- OWNER_ROLE (in module google.cloud.pubsub.iam), 105
- P**
- page_token (google.cloud.bigquery.query.QueryResults attribute), 140
- pan (google.cloud.vision.face.Angles attribute), 287
- parent (google.cloud.datastore.key.Key attribute), 40
- parent_name (google.cloud.monitoring.group.Group attribute), 255
- parse_message() (google.cloud.logging.entries.ProtobufEntry method), 219
- PARTICIPLE (google.cloud.language.syntax.PartOfSpeech attribute), 313
- partition_expiration (google.cloud.bigquery.table.Table attribute), 136
- partitioning_type (google.cloud.bigquery.table.Table attribute), 136
- PartOfSpeech (class in google.cloud.language.syntax), 312
- patch() (google.cloud.bigquery.dataset.Dataset method), 125
- patch() (google.cloud.bigquery.table.Table method), 136
- path (google.cloud.bigquery.dataset.Dataset attribute), 126
- path (google.cloud.bigquery.table.Table attribute), 136
- path (google.cloud.datastore.key.Key attribute), 40
- path (google.cloud.dns.changes.Changes attribute), 200
- path (google.cloud.dns.zone.ManagedZone attribute), 195
- path (google.cloud.logging.logger.Logger attribute), 218
- path (google.cloud.logging.metric.Metric attribute), 222
- path (google.cloud.logging.sink.Sink attribute), 224
- path (google.cloud.monitoring.group.Group attribute), 255
- path (google.cloud.pubsub.subscription.Subscription attribute), 98
- path (google.cloud.resource_manager.project.Project attribute), 184
- path (google.cloud.storage.blob.Blob attribute), 65
- path (google.cloud.storage.bucket.Bucket attribute), 74
- path_helper() (google.cloud.storage.blob.Blob static method), 66
- path_helper() (google.cloud.storage.bucket.Bucket static method), 74
- PERSON (google.cloud.language.entity.EntityType attribute), 312
- pixel_fraction (google.cloud.vision.color.ColorInformation attribute), 280
- PLAIN_TEXT (google.cloud.language.document.Document attribute), 308
- Point (class in google.cloud.monitoring.timeseries), 263
- Policy (class in google.cloud.pubsub.iam), 106
- poll() (google.cloud.operation.Operation method), 27
- Position (class in google.cloud.vision.geometry), 295
- position (google.cloud.vision.face.Landmark attribute), 291
- POSSIBLE (google.cloud.vision.likelihood.Likelihood attribute), 297
- PreconditionFailed, 14
- PREDEFINED_JSON_ACLS (google.cloud.storage.acl.ACL attribute),

- 78
 - preserve_nulls (google.cloud.bigquery.query.QueryResults attribute), 140
 - print_header (google.cloud.bigquery.job.ExtractTableToStorageJob attribute), 128
 - priority (google.cloud.bigquery.job.QueryJob attribute), 131
 - Project (class in google.cloud.bigquery.client), 122
 - Project (class in google.cloud.resource_manager.project), 183
 - project (google.cloud.bigquery.dataset.Dataset attribute), 126
 - project (google.cloud.bigquery.query.QueryResults attribute), 141
 - project (google.cloud.bigquery.table.Table attribute), 136
 - project (google.cloud.datastore.batch.Batch attribute), 52
 - project (google.cloud.datastore.key.Key attribute), 41
 - project (google.cloud.datastore.query.Query attribute), 46
 - project (google.cloud.datastore.transaction.Transaction attribute), 49
 - project (google.cloud.dns.zone.ManagedZone attribute), 195
 - project (google.cloud.logging.logger.Logger attribute), 218
 - project (google.cloud.logging.metric.Metric attribute), 222
 - project (google.cloud.logging.sink.Sink attribute), 224
 - project (google.cloud.pubsub.subscription.Subscription attribute), 97
 - project (google.cloud.pubsub.topic.Topic attribute), 92
 - PROJECT (in module google.cloud.environment_vars), 17
 - project_number (google.cloud.storage.bucket.Bucket attribute), 74
 - projection (google.cloud.datastore.query.Query attribute), 46
 - PRONOUN (google.cloud.language.syntax.PartOfSpeech attribute), 313
 - ProtobufEntry (class in google.cloud.logging.entries), 219
 - public_url (google.cloud.storage.blob.Blob attribute), 66
 - publish() (google.cloud.pubsub.topic.Batch method), 95
 - publish() (google.cloud.pubsub.topic.Topic method), 93
 - publisher_api (google.cloud.pubsub.client.Client attribute), 88
 - PUBSUB_ADMIN_ROLE (in module google.cloud.pubsub.iam), 105
 - PUBSUB_API_HOST (in module google.cloud.pubsub.connection), 89
 - PUBSUB_EDITOR_ROLE (in module google.cloud.pubsub.iam), 105
 - PUBSUB_EMULATOR (in module google.cloud.environment_vars), 17
 - PUBSUB_PUBLISHER_ROLE (in module google.cloud.pubsub.iam), 105
 - PUBSUB_SUBSCRIBER_ROLE (in module google.cloud.pubsub.iam), 105
 - PUBSUB_SUBSCRIPTIONS_CONSUME (in module google.cloud.pubsub.iam), 106
 - PUBSUB_SUBSCRIPTIONS_CREATE (in module google.cloud.pubsub.iam), 106
 - PUBSUB_SUBSCRIPTIONS_DELETE (in module google.cloud.pubsub.iam), 106
 - PUBSUB_SUBSCRIPTIONS_GET (in module google.cloud.pubsub.iam), 106
 - PUBSUB_SUBSCRIPTIONS_GET_IAM_POLICY (in module google.cloud.pubsub.iam), 106
 - PUBSUB_SUBSCRIPTIONS_LIST (in module google.cloud.pubsub.iam), 106
 - PUBSUB_SUBSCRIPTIONS_SET_IAM_POLICY (in module google.cloud.pubsub.iam), 106
 - PUBSUB_SUBSCRIPTIONS_UPDATE (in module google.cloud.pubsub.iam), 106
 - PUBSUB_TOPICS_CONSUME (in module google.cloud.pubsub.iam), 105
 - PUBSUB_TOPICS_CREATE (in module google.cloud.pubsub.iam), 105
 - PUBSUB_TOPICS_DELETE (in module google.cloud.pubsub.iam), 105
 - PUBSUB_TOPICS_GET (in module google.cloud.pubsub.iam), 105
 - PUBSUB_TOPICS_GET_IAM_POLICY (in module google.cloud.pubsub.iam), 105
 - PUBSUB_TOPICS_LIST (in module google.cloud.pubsub.iam), 105
 - PUBSUB_TOPICS_SET_IAM_POLICY (in module google.cloud.pubsub.iam), 105
 - PUBSUB_VIEWER_ROLE (in module google.cloud.pubsub.iam), 105
 - pull() (google.cloud.pubsub.subscription.Subscription method), 99
 - PUNCTUATION (google.cloud.language.syntax.PartOfSpeech attribute), 313
 - put() (google.cloud.datastore.batch.Batch method), 52
 - put() (google.cloud.datastore.client.Client method), 31
 - put() (google.cloud.datastore.transaction.Transaction method), 49
 - put_multi() (google.cloud.datastore.client.Client method), 31
- ## Q
- Query (class in google.cloud.datastore.query), 43
 - Query (class in google.cloud.monitoring.query), 257
 - query() (google.cloud.datastore.client.Client method), 31
 - query() (google.cloud.monitoring.client.Client method), 245
 - QueryJob (class in google.cloud.bigquery.job), 130
 - QueryPriority (class in google.cloud.bigquery.job), 131

- QueryResults (class in google.cloud.bigquery.query), 139
- quotas() (google.cloud.dns.client.Client method), 191
- quote_character (google.cloud.bigquery.job.LoadTableFromBigQueryJob attribute), 130
- ## R
- red (google.cloud.vision.color.Color attribute), 279
- Redirection, 14
- reduce() (google.cloud.monitoring.query.Query method), 259
- Reducer (class in google.cloud.monitoring.query), 262
- reload() (google.cloud.bigquery.dataset.Dataset method), 126
- reload() (google.cloud.bigquery.table.Table method), 136
- reload() (google.cloud.dns.changes.Changes method), 200
- reload() (google.cloud.dns.zone.ManagedZone method), 195
- reload() (google.cloud.logging.metric.Metric method), 222
- reload() (google.cloud.logging.sink.Sink method), 224
- reload() (google.cloud.monitoring.group.Group method), 255
- reload() (google.cloud.pubsub.subscription.Subscription method), 98
- reload() (google.cloud.resource_manager.project.Project method), 184
- reload() (google.cloud.storage.acl.ACL method), 79
- reload_path (google.cloud.storage.acl.BucketACL attribute), 80
- reload_path (google.cloud.storage.acl.ObjectACL attribute), 80
- rename_blob() (google.cloud.storage.bucket.Bucket method), 75
- report() (google.cloud.error_reporting.client.Client method), 235
- report_exception() (google.cloud.error_reporting.client.Client method), 236
- RequestRangeNotSatisfiable, 14
- reset() (google.cloud.storage.acl.ACL method), 79
- Resource (class in google.cloud.monitoring.resource), 251
- resource() (google.cloud.monitoring.client.Client static method), 245
- resource_record_set() (google.cloud.dns.zone.ManagedZone method), 195
- ResourceDescriptor (class in google.cloud.monitoring.resource), 251
- ResourceRecordSet (class in google.cloud.dns.resource_record_set), 197
- results() (google.cloud.bigquery.job.QueryJob method), 131
- ResumeIncomplete, 14
- reverse() (google.cloud.language.syntax.PartOfSpeech class method), 313
- RIGHT_EYE_BOTTOM_BOUNDARY (google.cloud.vision.face.FaceLandmarkTypes attribute), 290
- RIGHT_EYE_LEFT_CORNER (google.cloud.vision.face.FaceLandmarkTypes attribute), 291
- RIGHT_EYE_PUPIL (google.cloud.vision.face.FaceLandmarkTypes attribute), 291
- RIGHT_EYE_RIGHT_CORNER (google.cloud.vision.face.FaceLandmarkTypes attribute), 291
- RIGHT_EYE_TOP_BOUNDARY (google.cloud.vision.face.FaceLandmarkTypes attribute), 291
- RIGHT_EYEBROW_UPPER_MIDPOINT (google.cloud.vision.face.FaceLandmarkTypes attribute), 290
- RIGHT_OF_LEFT_EYEBROW (google.cloud.vision.face.FaceLandmarkTypes attribute), 291
- RIGHT_OF_RIGHT_EYEBROW (google.cloud.vision.face.FaceLandmarkTypes attribute), 291
- roll (google.cloud.vision.face.Angles attribute), 287
- rollback() (google.cloud.datastore.batch.Batch method), 53
- rollback() (google.cloud.datastore.connection.Connection method), 34
- rollback() (google.cloud.datastore.transaction.Transaction method), 50
- rows (google.cloud.bigquery.query.QueryResults attribute), 141
- run() (google.cloud.bigquery.query.QueryResults method), 141
- run_async_query() (google.cloud.bigquery.client.Client method), 121
- run_query() (google.cloud.datastore.connection.Connection method), 34
- run_sync_query() (google.cloud.bigquery.client.Client method), 121
- ## S
- SAFE_SEARCH_DETECTION (google.cloud.vision.feature.FeatureTypes attribute), 286
- SafeSearchAnnotation (class in google.cloud.vision.safe), 299

- save() (google.cloud.storage.acl.ACL method), 79
- save_path (google.cloud.storage.acl.BucketACL attribute), 80
- save_path (google.cloud.storage.acl.ObjectACL attribute), 80
- save_predefined() (google.cloud.storage.acl.ACL method), 80
- schema (google.cloud.bigquery.job.LoadTableFromStorageJob attribute), 130
- schema (google.cloud.bigquery.query.QueryResults attribute), 141
- schema (google.cloud.bigquery.table.Table attribute), 137
- SchemaField (class in google.cloud.bigquery.schema), 143
- SCOPE (google.cloud.bigquery.connection.Connection attribute), 122
- SCOPE (google.cloud.connection.Connection attribute), 9
- SCOPE (google.cloud.datastore.connection.Connection attribute), 32
- SCOPE (google.cloud.dns.connection.Connection attribute), 192
- SCOPE (google.cloud.language.connection.Connection attribute), 306
- SCOPE (google.cloud.logging.connection.Connection attribute), 213
- SCOPE (google.cloud.monitoring.connection.Connection attribute), 247
- SCOPE (google.cloud.pubsub.connection.Connection attribute), 89
- SCOPE (google.cloud.resource_manager.connection.Connection attribute), 181
- SCOPE (google.cloud.storage.connection.Connection attribute), 60
- SCOPE (google.cloud.vision.connection.Connection attribute), 278
- score (google.cloud.vision.color.ColorInformation attribute), 280
- score (google.cloud.vision.entity.EntityAnnotation attribute), 284
- select_group() (google.cloud.monitoring.query.Query method), 260
- select_interval() (google.cloud.monitoring.query.Query method), 260
- select_metrics() (google.cloud.monitoring.query.Query method), 260
- select_projects() (google.cloud.monitoring.query.Query method), 261
- select_resources() (google.cloud.monitoring.query.Query method), 261
- self_link (google.cloud.bigquery.dataset.Dataset attribute), 126
- self_link (google.cloud.bigquery.table.Table attribute), 137
- self_link (google.cloud.storage.blob.Blob attribute), 66
- self_link (google.cloud.storage.bucket.Bucket attribute), 75
- send() (google.cloud.logging.handlers.transports.background_thread.Backgrounder method), 229
- send() (google.cloud.logging.handlers.transports.base.Transport method), 231
- send() (google.cloud.logging.handlers.transports.sync.SyncTransport method), 227
- Sentence (class in google.cloud.language.syntax), 313
- sentences (google.cloud.language.document.Annotations attribute), 307
- Sentiment (class in google.cloud.language.sentiment), 312
- sentiment (google.cloud.language.document.Annotations attribute), 307
- ServerError, 14
- service_account() (google.cloud.pubsub.iam.Policy static method), 106
- service_timestamp (google.cloud.pubsub.message.Message attribute), 103
- ServiceUnavailable, 14
- set_iam_policy() (google.cloud.pubsub.subscription.Subscription method), 100
- set_iam_policy() (google.cloud.pubsub.topic.Topic method), 94
- set_properties_from_api_repr() (google.cloud.resource_manager.project.Project method), 184
- setup_logging() (in google.cloud.logging.handlers.handlers module), 226
- Sink (class in google.cloud.logging.sink), 223
- sink() (google.cloud.logging.client.Client method), 213
- sinks_api (google.cloud.logging.client.Client attribute), 213
- size (google.cloud.storage.blob.Blob attribute), 66
- skip_leading_rows (google.cloud.bigquery.job.LoadTableFromStorageJob attribute), 130
- sorrow_likelihood (google.cloud.vision.face.Emotions attribute), 288
- source (google.cloud.vision.image.Image attribute), 294
- source_format (google.cloud.bigquery.job.LoadTableFromStorageJob attribute), 130
- SourceFormat (class in google.cloud.bigquery.job), 131
- spoof (google.cloud.vision.safe.SafeSearchAnnotation attribute), 299
- started (google.cloud.dns.changes.Changes attribute), 200
- status (google.cloud.dns.changes.Changes attribute), 200
- storage_class (google.cloud.storage.blob.Blob attribute), 66
- storage_class (google.cloud.storage.bucket.Bucket attribute), 75
- StructEntry (class in google.cloud.logging.entries), 219

- subscriber_api (google.cloud.pubsub.client.Client attribute), 88
 - Subscription (class in google.cloud.pubsub.subscription), 97
 - subscription() (google.cloud.pubsub.topic.Topic method), 91
 - surprise_likelihood (google.cloud.vision.face.Emotions attribute), 288
 - SyncTransport (class in google.cloud.logging.handlers.transports.sync), 227
- T**
- Table (class in google.cloud.bigquery.table), 133
 - table() (google.cloud.bigquery.dataset.Dataset method), 126
 - table_id (google.cloud.bigquery.table.Table attribute), 137
 - table_type (google.cloud.bigquery.table.Table attribute), 137
 - target (google.cloud.operation.Operation attribute), 27
 - TemporaryRedirect, 15
 - TEXT_DETECTION (google.cloud.vision.feature.FeatureType attribute), 286
 - TextEntry (class in google.cloud.logging.entries), 219
 - tilt (google.cloud.vision.face.Angles attribute), 287
 - time_created (google.cloud.storage.bucket.Bucket attribute), 75
 - time_deleted (google.cloud.storage.blob.Blob attribute), 66
 - time_series() (google.cloud.monitoring.client.Client static method), 246
 - timeout_ms (google.cloud.bigquery.query.QueryResults attribute), 141
 - TimeSeries (class in google.cloud.monitoring.timeseries), 263
 - timestamp (google.cloud.pubsub.message.Message attribute), 103
 - to_api_repr() (google.cloud.pubsub.iam.Policy method), 107
 - to_protobuf() (google.cloud.datastore.key.Key method), 41
 - Token (class in google.cloud.language.syntax), 314
 - tokens (google.cloud.language.document.Annotations attribute), 307
 - TooManyRequests, 15
 - Topic (class in google.cloud.pubsub.topic), 91
 - topic() (google.cloud.pubsub.client.Client method), 88
 - total_bytes_processed (google.cloud.bigquery.query.QueryResults attribute), 141
 - total_rows (google.cloud.bigquery.query.QueryResults attribute), 141
 - Transaction (class in google.cloud.datastore.transaction), 47
 - transaction() (google.cloud.datastore.client.Client method), 32
 - translate() (google.cloud.translate.client.Client method), 270
 - Transport (class in google.cloud.logging.handlers.transports.base), 231
 - TYPE_UNSPECIFIED (google.cloud.language.document.Document attribute), 308
- U**
- Unauthorized, 15
 - undelese() (google.cloud.resource_manager.project.Project method), 184
 - underexposed_likelihood (google.cloud.vision.face.FaceImageProperties attribute), 290
 - UNKNOWN (google.cloud.language.entity.EntityType attribute), 312
 - UNKNOWN (google.cloud.language.syntax.PartOfSpeech attribute), 313
 - UNKNOWN (google.cloud.vision.likelihood.Likelihood attribute), 297
 - UNKNOWN_LANDMARK (google.cloud.vision.face.FaceLandmarkTypes attribute), 291
 - UNLIKELY (google.cloud.vision.likelihood.Likelihood attribute), 297
 - update() (google.cloud.bigquery.dataset.Dataset method), 126
 - update() (google.cloud.bigquery.table.Table method), 137
 - update() (google.cloud.logging.metric.Metric method), 222
 - update() (google.cloud.logging.sink.Sink method), 224
 - update() (google.cloud.monitoring.group.Group method), 256
 - update() (google.cloud.resource_manager.project.Project method), 185
 - updated (google.cloud.storage.blob.Blob attribute), 66
 - upload_from_file() (google.cloud.bigquery.table.Table method), 137
 - upload_from_file() (google.cloud.storage.blob.Blob method), 67
 - upload_from_filename() (google.cloud.storage.blob.Blob method), 67
 - upload_from_string() (google.cloud.storage.blob.Blob method), 68
 - UPPER_LIP (google.cloud.vision.face.FaceLandmarkTypes attribute), 291
 - use_legacy_sql (google.cloud.bigquery.job.QueryJob attribute), 131
 - use_legacy_sql (google.cloud.bigquery.query.QueryResults attribute), 141
 - use_query_cache (google.cloud.bigquery.job.QueryJob attribute), 131

use_query_cache (google.cloud.bigquery.query.QueryResults attribute), 141

user() (google.cloud.pubsub.iam.Policy static method), 106

user() (google.cloud.storage.acl.ACL method), 80

UTF16 (google.cloud.language.document.Encoding attribute), 309

UTF32 (google.cloud.language.document.Encoding attribute), 309

UTF8 (google.cloud.language.document.Encoding attribute), 309

V

VALUE_TYPE_UNSPECIFIED

(google.cloud.monitoring.metric.ValueType attribute), 250

ValueType (class in google.cloud.monitoring.metric), 250

VERB (google.cloud.language.syntax.PartOfSpeech attribute), 313

versioning_enabled (google.cloud.storage.bucket.Bucket attribute), 75

Vertex (class in google.cloud.vision.geometry), 296

vertices (google.cloud.vision.geometry.BoundsBase attribute), 295

VERY_LIKELY (google.cloud.vision.likelihood.Likelihood attribute), 297

VERY_UNLIKELY (google.cloud.vision.likelihood.Likelihood attribute), 297

view_query (google.cloud.bigquery.table.Table attribute), 138

VIEWER_ROLE (in module google.cloud.pubsub.iam), 105

violence (google.cloud.vision.safe.SafeSearchAnnotation attribute), 300

VisionRequest (class in google.cloud.vision.client), 278

W

WORK_OF_ART (google.cloud.language.entity.EntityType attribute), 312

write_disposition (google.cloud.bigquery.job.CopyJob attribute), 127

write_disposition (google.cloud.bigquery.job.LoadTableFromStorageJob attribute), 130

write_disposition (google.cloud.bigquery.job.QueryJob attribute), 131

WriteDisposition (class in google.cloud.bigquery.job), 131

X

x_coordinate (google.cloud.vision.geometry.Position attribute), 296

x_coordinate (google.cloud.vision.geometry.Vertex attribute), 296

y_coordinate (google.cloud.vision.geometry.Position attribute), 296

y_coordinate (google.cloud.vision.geometry.Vertex attribute), 296

Z

z_coordinate (google.cloud.vision.geometry.Position attribute), 296

zone() (google.cloud.dns.client.Client method), 191

zone_id (google.cloud.dns.zone.ManagedZone attribute), 195