googledatastore Documentation

Release

Author

March 26, 2016

Contents

1 googledatastore Package **3**
- 1.1 `googledatastore` Package . . . 3
- 1.2 `connection` Module . . . 3
- 1.3 `datastore_v1_pb2` Module . . . 5
- 1.4 `helper` Module . . . 5

2 Indices and tables **9**

Python Module Index **11**

Contents:

CHAPTER 1

googledatastore Package

1.1 googledatastore Package

googledatastore client.

googledatastore.__init__.**allocate_ids**(*request*)
See connection.Datastore.allocate_ids.

googledatastore.__init__.**begin_transaction**(*request*)
See connection.Datastore.begin_transaction.

googledatastore.__init__.**commit**(*request*)
See connection.Datastore.commit.

googledatastore.__init__.**get_default_connection**()
Return the default datastore connection.

dataset defaults to helper.get_dataset_from_env(), host to os.getenv('DATASTORE_HOST'), and credentials to helper.get_credentials_from_env().

Use set_options to override defaults.

googledatastore.__init__.**lookup**(*request*)
See connection.Datastore.lookup.

googledatastore.__init__.**rollback**(*request*)
See connection.Datastore.rollback.

googledatastore.__init__.**run_query**(*request*)
See connection.Datastore.run_query.

googledatastore.__init__.**set_options**(***kwargs*)
Set datastore connection options.

Args: credentials: oauth2client.Credentials to authorize the connection. dataset: the dataset to send RPCs to. host: the host used to construct the datastore API, default to Google APIs production server.

1.2 connection Module

googledatastore connection.

class googledatastore.connection.**Datastore**(*dataset*, *credentials=None*, *host=None*)
Bases: object

Datastore client connection constructor.

allocate_ids(*request*)
Allocate ids for incomplete keys.

Args: request: AllocateIdsRequest proto message.

Returns: AllocateIdsResponse proto message.

Raises: RPCError: The underlying RPC call failed with an HTTP error. (See: .response attribute)

begin_transaction(*request*)
Begin a new transaction.

Args: request: BeginTransactionRequest proto message.

Returns: BeginTransactionResponse proto message.

Raises: RPCError: The underlying RPC call failed with an HTTP error. (See: .response attribute)

commit(*request*)
Commit a mutation, transaction or mutation in a transaction.

Args: request: CommitRequest proto message.

Returns: CommitResponse proto message.

Raises: RPCError: The underlying RPC call failed with an HTTP error. (See: .response attribute)

lookup(*request*)
Lookup entities by key.

Args: request: LookupRequest proto message.

Returns: LookupResponse proto message.

Raises: RPCError: The underlying RPC call failed with an HTTP error. (See: .response attribute)

rollback(*request*)
Rollback a transaction.

Args: request: RollbackRequest proto message.

Returns: RollbackResponse proto message.

Raises: RPCError: The underlying RPC call failed with an HTTP error. (See: .response attribute)

run_query(*request*)
Query for entities.

Args: request: RunQueryRequest proto message.

Returns: RunQueryResponse proto message.

Raises: RPCError: The underlying RPC call failed with an HTTP error. (See: .response attribute)

exception googledatastore.connection.**Error**
Bases: exceptions.Exception

A Datastore service error occured.

exception googledatastore.connection.**HTTPError**
Bases: *googledatastore.connection.Error*

An HTTP error occured.

response = None

exception googledatastore.connection.**RPCError**(*method*, *response*, *content*)
Bases: *googledatastore.connection.HTTPError*

The Datastore RPC failed.

method = None

reason = None

exception googledatastore.connection.**AuthError**
Bases: *googledatastore.connection.Error*

Authentication failed.

exception googledatastore.connection.**BadArgumentError**
Bases: *googledatastore.connection.Error*

Argument validation failed.

1.3 datastore_v1_pb2 Module

1.4 helper Module

googledatastore helper.

googledatastore.helper.**get_credentials_from_env**()
Get datastore credentials from the environment.

Try and fallback on the following credentials in that order: - Google APIs Signed JWT credentials based on DATASTORE_SERVICE_ACCOUNT and DATASTORE_PRIVATE_KEY_FILE environment variables - Compute Engine service account - No credentials (development server)

Returns: datastore credentials.

googledatastore.helper.**add_key_path**(*key_proto*, **path_elements*)
Add path elements to the given datastore.Key proto message.

Args: key_proto: datastore.Key proto message. path_elements: list of ancestors to add to the key. (kind1, id1/name1, ..., kindN, idN/nameN), the last 2 elements represent the entity key, if no terminating id/name: they key will be an incomplete key.

Raises: TypeError: the given id or name has the wrong type.

Returns: the same datastore.Key.

Usage:

```
>>> add_key_path(key_proto, 'Kind', 'name')  # no parent, with name
datastore.Key(...)
>>> add_key_path(key_proto, 'Kind2', 1)  # no parent, with id
datastore.Key(...)
>>> add_key_path(key_proto, 'Kind', 'name', 'Kind2', 1)  # parent, complete
datastore.Key(...)
>>> add_key_path(key_proto, 'Kind', 'name', 'Kind2')  # parent, incomplete
datastore.Key(...)
```

googledatastore.helper.**add_properties**(*entity_proto*, *property_dict*, *indexed=None*)
Add values to the given datastore.Entity proto message.

Args: entity_proto: datastore.Entity proto message. property_dict: a dictionary from property name to either a python object or

datastore.Value.

indexed: if the property values should be indexed. None leaves indexing as is (defaults to True if value is a python object).

Usage:

```
>>> add_properties(proto, {'foo': u'a', 'bar': [1, 2]})
```

Raises: TypeError: if a given property value type is not supported.

googledatastore.helper.**set_property**(*property_proto*, *name*, *value*, *indexed=None*)
Set property value in the given datastore.Property proto message.

Args: property_proto: datastore.Property proto message. name: name of the property. value: python object or datastore.Value. indexed: if the value should be indexed. None leaves indexing as is

(defaults to True if value is a python object).

Usage:

```
>>> set_property(property_proto, 'foo', u'a')
```

Raises: TypeError: if the given value type is not supported.

googledatastore.helper.**set_value**(*value_proto*, *value*, *indexed=None*)
Set the corresponding datastore.Value _value field for the given arg.

Args: value_proto: datastore.Value proto message. value: python object or datastore.Value. (unicode value will set a

datastore string value, str value will set a blob string value). Undefined behavior if value is/contains value_proto.

indexed: if the value should be indexed. None leaves indexing as is (defaults to True if value is not a Value message).

Raises: TypeError: if the given value type is not supported.

googledatastore.helper.**get_value**(*value_proto*)
Gets the python object equivalent for the given value proto.

Args: value_proto: datastore.Value proto message.

Returns: the corresponding python object value. timestamps are converted to datetime, and datastore.Value is returned for blob_key_value.

googledatastore.helper.**get_property_dict**(*entity_proto*)
Convert datastore.Entity to a dict of property name -> datastore.Value.

Args: entity_proto: datastore.Entity proto message.

Usage:

```
>>> get_property_dict(entity_proto)
{'foo': {string_value='a'}, 'bar': {integer_value=2}}
```

Returns: dict of entity properties.

googledatastore.helper.**set_kind**(*query_proto*, *kind*)
Set the kind constraint for the given datastore.Query proto message.

googledatastore.helper.**add_property_orders**(*query_proto*, **orders*)
Add ordering constraint for the given datastore.Query proto message.

Args: query_proto: datastore.Query proto message. orders: list of propertype name string, default to ascending order and set descending if prefixed by '-'.

Usage:

```
>>> add_property_orders(query_proto, 'foo')  # sort by foo asc
>>> add_property_orders(query_proto, '-bar')  # sort by bar desc
```

googledatastore.helper.**add_projection**(*query_proto*, **projection*)
Add projection properties to the given datatstore.Query proto message.

googledatastore.helper.**set_property_filter**(*filter_proto*, *name*, *op*, *value*)
Set property filter contraint in the given datastore.Filter proto message.

Args: filter_proto: datastore.Filter proto message name: property name op: datastore.PropertyFilter.Operation value: property value

Returns: the same datastore.Filter.

Usage:

```
>>> set_property_filter(filter_proto, 'foo',
...   datastore.PropertyFilter.EQUAL, 'a')  # WHERE 'foo' = 'a'
```

googledatastore.helper.**set_composite_filter**(*filter_proto*, *op*, **filters*)
Set composite filter contraint in the given datastore.Filter proto message.

Args: filter_proto: datastore.Filter proto message op: datastore.CompositeFilter.Operation filters: vararg list of datastore.Filter

Returns: the same datastore.Filter.

Usage:

```
>>> set_composite_filter(filter_proto, datastore.CompositeFilter.AND,
...   set_property_filter(datastore.Filter(), ...),
...   set_property_filter(datastore.Filter(), ...)) # WHERE ... AND ...
```

googledatastore.helper.**to_timestamp_usec**(*dt*)
Convert datetime to microsecond timestamp.

Args: dt: a timezone naive datetime.

Returns: a microsecond timestamp as a long.

Raises: TypeError: if a timezone aware datetime was provided.

googledatastore.helper.**from_timestamp_usec**(*timestamp*)
Convert microsecond timestamp to datetime.

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

g

googledatastore.__init__, 3
googledatastore.connection, 3
googledatastore.datastore_v1_pb2, 5
googledatastore.helper, 5

A

add_key_path() (in module googledatastore.helper), 5
add_projection() (in module googledatastore.helper), 7
add_properties() (in module googledatastore.helper), 5
add_property_orders() (in module googledatastore.helper), 6
allocate_ids() (googledatastore.connection.Datastore method), 4
allocate_ids() (in module googledatastore.__init__), 3
AuthError, 5

B

BadArgumentError, 5
begin_transaction() (googledatastore.connection.Datastore method), 4
begin_transaction() (in module googledatastore.__init__), 3

C

commit() (googledatastore.connection.Datastore method), 4
commit() (in module googledatastore.__init__), 3

D

Datastore (class in googledatastore.connection), 3

E

Error, 4

F

from_timestamp_usec() (in module googledatastore.helper), 7

G

get_credentials_from_env() (in module googledatastore.helper), 5
get_default_connection() (in module googledatastore.__init__), 3
get_property_dict() (in module googledatastore.helper), 6
get_value() (in module googledatastore.helper), 6
googledatastore.__init__ (module), 3
googledatastore.connection (module), 3
googledatastore.datastore_v1_pb2 (module), 5
googledatastore.helper (module), 5

H

HTTPError, 4

L

lookup() (googledatastore.connection.Datastore method), 4
lookup() (in module googledatastore.__init__), 3

M

method (googledatastore.connection.RPCError attribute), 5

R

reason (googledatastore.connection.RPCError attribute), 5
response (googledatastore.connection.HTTPError attribute), 4
rollback() (googledatastore.connection.Datastore method), 4
rollback() (in module googledatastore.__init__), 3
RPCError, 4
run_query() (googledatastore.connection.Datastore method), 4
run_query() (in module googledatastore.__init__), 3

S

set_composite_filter() (in module googledatastore.helper), 7
set_kind() (in module googledatastore.helper), 6
set_options() (in module googledatastore.__init__), 3
set_property() (in module googledatastore.helper), 6
set_property_filter() (in module googledatastore.helper), 7
set_value() (in module googledatastore.helper), 6

T

to_timestamp_usec() (in module googledatastore.helper),
7