

---

**influxdb***client*

***Release 1.0.0***

**Nov 11, 2019**



---

## Contents:

---

<b>1</b>	<b>User Guide</b>	<b>1</b>
1.1	Query . . . . .	1
1.2	Write . . . . .	2
1.2.1	The data could be written as . . . . .	2
1.2.2	Batching . . . . .	2
1.2.3	Asynchronous client . . . . .	4
1.2.4	Synchronous client . . . . .	4
1.3	Queries . . . . .	4
1.4	Examples . . . . .	6
1.4.1	How to efficiently import large dataset . . . . .	6
1.5	Gzip support . . . . .	8
1.6	Debugging . . . . .	8
<b>2</b>	<b>API Reference</b>	<b>9</b>
2.1	InfluxDBClient . . . . .	9
2.2	QueryApi . . . . .	11
2.3	WriteApi . . . . .	12
2.4	BucketsApi . . . . .	12
2.5	LabelsApi . . . . .	14
2.6	OrganizationsApi . . . . .	15
2.7	UsersApi . . . . .	16
2.8	TasksApi . . . . .	17
<b>3</b>	<b>InfluxDB 2.0 client features</b>	<b>23</b>
<b>4</b>	<b>Installation</b>	<b>25</b>
4.1	pip install . . . . .	25
4.2	Setuptools . . . . .	25
<b>5</b>	<b>Getting Started</b>	<b>27</b>
<b>6</b>	<b>Indices and tables</b>	<b>29</b>
	<b>Index</b>	<b>31</b>



- *Query*
- *Write*
  - *The data could be written as*
  - *Batching*
  - *Asynchronous client*
  - *Synchronous client*
- *Queries*
- *Examples*
  - *How to efficiently import large dataset*
- *Gzip support*
- *Debugging*

## 1.1 Query

```
from influxdb_client import InfluxDBClient, Point
from influxdb_client.client.write_api import SYNCHRONOUS

bucket = "my-bucket"

client = InfluxDBClient(url="http://localhost:9999", token="my-token", org="my-org")

write_api = client.write_api(write_options=SYNCHRONOUS)
query_api = client.query_api()
```

(continues on next page)

(continued from previous page)

```

p = Point("my_measurement").tag("location", "Prague").field("temperature", 25.3)

write_api.write(bucket=bucket, org="my-org", record=p)

## using Table structure
tables = query_api.query('from(bucket:"my-bucket") |> range(start: -10m)')

for table in tables:
    print(table)
    for row in table.records:
        print(row.values)

## using csv library
csv_result = query_api.query_csv('from(bucket:"my-bucket") |> range(start: -10m)')
val_count = 0
for row in csv_result:
    for cell in row:
        val_count += 1

```

## 1.2 Write

The `WriteApi` supports synchronous, asynchronous and batching writes into InfluxDB 2.0. The data should be passed as an `InfluxDB Line Protocol`, `Data Point` or `Observable` stream.

*The default instance of `WriteApi` use batching.*

### 1.2.1 The data could be written as

1. string or bytes that is formatted as an InfluxDB's line protocol
2. `Data Point` structure
3. Dictionary style mapping with keys: measurement, tags, fields and time
4. List of above items
5. A batching type of write also supports an `Observable` that produce one of an above item

### 1.2.2 Batching

The batching is configurable by `write_options`:

Property	Description	Default Value
<b>batch_size</b>	the number of data points to collect in a batch	1000
<b>flush_interval</b>	the number of milliseconds before the batch is written	1000
<b>jitter_interval</b>	the number of milliseconds to increase the batch flush interval by a random amount	0
<b>retry_interval</b>	the number of milliseconds to retry unsuccessful write. The retry interval is used when the InfluxDB server does not specify "Retry-After" header.	1000

```

import rx
from rx import operators as ops

from influxdb_client import InfluxDBClient, Point, WriteOptions
from influxdb_client.client.write_api import SYNCHRONOUS

_client = InfluxDBClient(url="http://localhost:9999", token="my-token", org="my-org")
_write_client = _client.write_api(write_options=WriteOptions(batch_size=500,
                                                             flush_interval=10_000,
                                                             jitter_interval=2_000,
                                                             retry_interval=5_000))

"""
Write Line Protocol formatted as string
"""
_write_client.write("my-bucket", "my-org", "h2o_feet,location=coyote_creek water_
↪level=1.0 1")
_write_client.write("my-bucket", "my-org", ["h2o_feet,location=coyote_creek water_
↪level=2.0 2",
                                           "h2o_feet,location=coyote_creek water_
↪level=3.0 3"])

"""
Write Line Protocol formatted as byte array
"""
_write_client.write("my-bucket", "my-org", "h2o_feet,location=coyote_creek water_
↪level=1.0 1".encode())
_write_client.write("my-bucket", "my-org", ["h2o_feet,location=coyote_creek water_
↪level=2.0 2".encode(),
                                           "h2o_feet,location=coyote_creek water_
↪level=3.0 3".encode()])

"""
Write Dictionary-style object
"""
_write_client.write("my-bucket", "my-org", {"measurement": "h2o_feet", "tags": {
↪"location": "coyote_creek"},
                                           "fields": {"water_level": 1.0}, "time": 1}
↪)
_write_client.write("my-bucket", "my-org", [{"measurement": "h2o_feet", "tags": {
↪"location": "coyote_creek"},
                                           "fields": {"water_level": 2.0}, "time": 2}
↪,
                                           {"measurement": "h2o_feet", "tags": {
↪"location": "coyote_creek"},
                                           "fields": {"water_level": 3.0}, "time": 3}
↪])

"""
Write Data Point
"""
_write_client.write("my-bucket", "my-org", Point("h2o_feet").tag("location", "coyote_
↪creek").field("water_level", 4.0).time(4))
_write_client.write("my-bucket", "my-org", [Point("h2o_feet").tag("location", "coyote_
↪creek").field("water_level", 5.0).time(5),
                                           Point("h2o_feet").tag("location", "coyote_
↪creek").field("water_level", 6.0).time(6)])

```

(continues on next page)

(continued from previous page)

```

"""
Write Observable stream
"""
_data = rx \
    .range(7, 11) \
    .pipe(ops.map(lambda i: "h2o_feet,location=coyote_creek water_level={0}.0 {0}".
    ↪format(i)))

_write_client.write("my-bucket", "my-org", _data)

"""
Close client
"""
_write_client.__del__()
_client.__del__()

```

### 1.2.3 Asynchronous client

Data are writes in an asynchronous HTTP request.

```

from influxdb_client import InfluxDBClient
from influxdb_client.client.write_api import ASYNCHRONOUS

client = InfluxDBClient(url="http://localhost:9999", token="my-token", org="my-org")
write_client = client.write_api(write_options=ASYNCHRONOUS)

...

client.__del__()

```

### 1.2.4 Synchronous client

Data are writes in a synchronous HTTP request.

```

from influxdb_client import InfluxDBClient
from influxdb_client.client.write_api import SYNCHRONOUS

client = InfluxDBClient(url="http://localhost:9999", token="my-token", org="my-org")
write_client = client.write_api(write_options=SYNCHRONOUS)

...

client.__del__()

```

## 1.3 Queries

The result retrieved by `QueryApi` could be formatted as a:

1. Flux data structure: `FluxTable`, `FluxColumn` and `FluxRecord`



2. `csv.reader` which will iterate over CSV lines
3. Raw unprocessed results as a `str` iterator

The API also support streaming FluxRecord via `query_stream`, see example below:

```
from influxdb_client import InfluxDBClient, Point, Dialect
from influxdb_client.client.write_api import SYNCHRONOUS

client = InfluxDBClient(url="http://localhost:9999", token="my-token", org="my-org")

write_api = client.write_api(write_options=SYNCHRONOUS)
query_api = client.query_api()

"""
Prepare data
"""

_point1 = Point("my_measurement").tag("location", "Prague").field("temperature", 25.3)
_point2 = Point("my_measurement").tag("location", "New York").field("temperature", 24.
↪3)

write_api.write(bucket="my-bucket", org="my-org", record=[_point1, _point2])

"""
Query: using Table structure
"""
tables = query_api.query('from(bucket:"my-bucket") |> range(start: -10m)')

for table in tables:
    print(table)
    for record in table.records:
        print(record.values)

print()
print()

"""
Query: using Stream
"""
records = query_api.query_stream('from(bucket:"my-bucket") |> range(start: -10m)')

for record in records:
    print(f'Temperature in {record["location"]} is {record["_value"]}')

"""
Interrupt a stream after retrieve a required data
"""
large_stream = query_api.query_stream('from(bucket:"my-bucket") |> range(start: -100d)
↪')
for record in large_stream:
    if record["location"] == "New York":
        print(f'New York temperature: {record["_value"]}')
        break

large_stream.close()

print()
print()
```

(continues on next page)

(continued from previous page)

```

"""
Query: using csv library
"""
csv_result = query_api.query_csv('from(bucket:"my-bucket") |> range(start: -10m)',
                                dialect=Dialect(header=False, delimiter=",", comment_
↪ prefix="#", annotations=[],
                                date_time_format="RFC3339"))

for csv_line in csv_result:
    if not len(csv_line) == 0:
        print(f'Temperature in {csv_line[9]} is {csv_line[6]}')

"""
Close client
"""
client.__del__()

```

## 1.4 Examples

### 1.4.1 How to efficiently import large dataset

- sources - import\_data\_set.py

```

"""
Import VIX - CBOE Volatility Index - from "vix-daily.csv" file into InfluxDB 2.0
https://datahub.io/core/finance-vix#data
"""

from collections import OrderedDict
from csv import DictReader
from datetime import datetime

import rx
from rx import operators as ops

from influxdb_client import InfluxDBClient, Point, WriteOptions

def parse_row(row: OrderedDict):
    """Parse row of CSV file into Point with structure:

        financial-analysis,type=ily close=18.47,high=19.82,low=18.28,open=19.82_
↪11981952000000000000

    CSV format:
    Date,VIX Open,VIX High,VIX Low,VIX Close\n
    2004-01-02,17.96,18.68,17.54,18.22\n
    2004-01-05,18.45,18.49,17.44,17.49\n
    2004-01-06,17.66,17.67,16.19,16.73\n
    2004-01-07,16.72,16.75,15.5,15.5\n
    2004-01-08,15.42,15.68,15.32,15.61\n
    2004-01-09,16.15,16.88,15.57,16.75\n
    ...
    """

```

(continues on next page)

(continued from previous page)

```

:param row: the row of CSV file
:return: Parsed csv row to [Point]
"""
return Point("financial-analysis") \
    .tag("type", "vix-daily") \
    .field("open", float(row['VIX Open'])) \
    .field("high", float(row['VIX High'])) \
    .field("low", float(row['VIX Low'])) \
    .field("close", float(row['VIX Close'])) \
    .time(datetime.strptime(row['Date'], '%Y-%m-%d'))

"""
Converts vix-daily.csv into sequence of datad point
"""
data = rx \
    .from_iterable(DictReader(open('vix-daily.csv', 'r'))) \
    .pipe(ops.map(lambda row: parse_row(row)))

client = InfluxDBClient(url="http://localhost:9999", token="my-token", org="my-org",
    debug=True)

"""
Create client that writes data in batches with 500 items.
"""
write_api = client.write_api(write_options=WriteOptions(batch_size=500, jitter_
    interval=1_000))

"""
Write data into InfluxDB
"""
write_api.write(org="my-org", bucket="my-bucket", record=data)
write_api.__del__()

"""
Querying max value of CBOE Volatility Index
"""
query = 'from(bucket:"my-bucket")' \
    ' |> range(start: 0, stop: now())' \
    ' |> filter(fn: (r) => r._measurement == "financial-analysis")' \
    ' |> max()'
result = client.query_api().query(org="my-org", query=query)

"""
Processing results
"""
print()
print("=== results ===")
print()
for table in result:
    for record in table.records:
        print('max {0:5} = {1}'.format(record.get_field(), record.get_value()))

"""
Close client
"""
client.__del__()

```

## 1.5 Gzip support

`InfluxDBClient` does not enable gzip compression for http requests by default. If you want to enable gzip to reduce transfer data's size, you can call:

```
from influxdb_client import InfluxDBClient

_db_client = InfluxDBClient(url="http://localhost:9999", token="my-token", org="my-org",
↪enable_gzip=True)
```

## 1.6 Debugging

For debug purpose you can enable verbose logging of http requests. Both request header and body will be logged to standard output.

```
_client = InfluxDBClient(url="http://localhost:9999", token="my-token", debug=True,
↪org="my-org")
```

- *InfluxDBClient*
- *QueryApi*
- *WriteApi*
- *BucketsApi*
- *LabelsApi*
- *OrganizationsApi*
- *UsersApi*
- *TasksApi*

## 2.1 InfluxDBClient

**class** `influxdb_client.InfluxDBClient` (*url*, *token*, *debug=None*, *timeout=10000*, *enable\_gzip=False*, *org: str = None*)  
*influxdb\_client.InfluxDBClient* is client for HTTP API defined in <https://github.com/influxdata/influxdb/blob/master/http/swagger.yml>.

### Parameters

- **url** – InfluxDB server API url (ex. <http://localhost:9999>).
- **token** – auth token
- **debug** – enable verbose logging of http requests
- **timeout** – default http client timeout
- **enable\_gzip** – Enable Gzip compression for http requests. Currently only the “Write” and “Query” endpoints supports the Gzip compression.

- **org** – organization name (used as a default in query and write API)

**authorizations\_api** () → influxdb\_client.client.authorizations\_api.AuthorizationsApi  
Creates the Authorizations API instance.

**Returns** authorizations api

**buckets\_api** () → influxdb\_client.client.bucket\_api.BucketsApi  
Creates the Bucket API instance.

**Returns** buckets api

**close** ()  
Shutdowns the client

**delete\_api** () → influxdb\_client.client.delete\_api.DeleteApi  
Gets the delete metrics API instance :return: delete api

**health** () → influxdb\_client.domain.health\_check.HealthCheck  
Get the health of an instance.

**Returns** HealthCheck

**labels\_api** () → influxdb\_client.client.labels\_api.LabelsApi  
Creates the Labels API instance.

**Returns** labels api

**organizations\_api** () → influxdb\_client.client.organizations\_api.OrganizationsApi  
Creates the Organizations API instance.

**Returns** organizations api

**query\_api** () → influxdb\_client.client.query\_api.QueryApi  
Creates a Query API instance

**Returns** Query api instance

**ready** () → influxdb\_client.domain.ready.Ready  
Gets The readiness of the InfluxDB 2.0.

**Returns** Ready

**tasks\_api** () → influxdb\_client.client.tasks\_api.TasksApi  
Creates the Tasks API instance.

**Returns** tasks api

**users\_api** () → influxdb\_client.client.users\_api.UsersApi  
Creates the Users API instance.

**Returns** users api

**write\_api** (write\_options=<influxdb\_client.client.write\_api.WriteOptions object>) → influxdb\_client.client.write\_api.WriteApi  
Creates a Write API instance

**Parameters** **write\_options** – write api configuration

**Returns** write api instance

## 2.2 QueryApi

**class** `influxdb_client.QueryApi` (`influxdb_client`)

Initializes query client.

**Parameters** `influxdb_client` – influxdb client

**query** (`query: str, org=None`) → `List[influxdb_client.client.flux_table.FluxTable]`

Synchronously executes the Flux query and return result as a `List['FluxTable']`

**Parameters**

- **query** – the Flux query
- **org** – organization name (optional if already specified in `InfluxDBClient`)

**Returns**

**query\_csv** (`query: str, org=None, dialect: influxdb_client.domain.dialect.Dialect = {'annotations': ['datatype', 'group', 'default'], 'comment_prefix': '#', 'date_time_format': 'RFC3339', 'delimiter': ',', 'header': True}`)

Executes the Flux query and return results as a CSV iterator. Each iteration returns a row of the CSV file.

**Parameters**

- **query** – a Flux query
- **org** – organization name (optional if already specified in `InfluxDBClient`)
- **dialect** – csv dialect format

**Returns** The returned object is an iterator. Each iteration returns a row of the CSV file (which can span multiple input lines).

**query\_raw** (`query: str, org=None, dialect={'annotations': ['datatype', 'group', 'default'], 'comment_prefix': '#', 'date_time_format': 'RFC3339', 'delimiter': ',', 'header': True}`)

Synchronously executes the Flux query and return result as raw unprocessed result as a `str`

**Parameters**

- **query** – a Flux query
- **org** – organization name (optional if already specified in `InfluxDBClient`)
- **dialect** – csv dialect format

**Returns** `str`

**query\_stream** (`query: str, org=None`) → `Generator[[influxdb_client.client.flux_table.FluxRecord, Any], None]`

Synchronously executes the Flux query and return stream of `FluxRecord` as a `Generator['FluxRecord']`

**Parameters**

- **query** – the Flux query
- **org** – organization name (optional if already specified in `InfluxDBClient`)

**Returns**

## 2.3 WriteApi

```
class influxdb_client.WriteApi (influxdb_client, write_options: influxdb_client.client.write_api.WriteOptions = influxdb_client.client.write_api.WriteOptions object>)
```

```
write (bucket: str, org: str, record: Union[str, List[str], influxdb_client.client.write.point.Point, List[Point], dict, List[dict], bytes, List[bytes], rx.core.observable.observable.Observable], write_precision: influxdb_client.domain.write_precision.WritePrecision = 'ns') → None
Writes time-series data into influxdb.
```

### Parameters

- **org** (*str*) – specifies the destination organization for writes; take either the ID or Name interchangeably; if both orgID and org are specified, org takes precedence. (required)
- **bucket** (*str*) – specifies the destination bucket for writes (required)
- **write\_precision** (*WritePrecision*) – specifies the precision for the unix timestamps within the body line-protocol
- **record** – Points, line protocol, RxPY Observable to write

## 2.4 BucketsApi

```
class influxdb_client.BucketsApi (influxdb_client)
```

```
create_bucket (bucket=None, bucket_name=None, org_id=None, retention_rules=None, description=None) → influxdb_client.domain.bucket.Bucket
Create a bucket # noqa: E501
```

### Parameters

- **bucket** (*Bucket*) – bucket to create (required)
- **bucket\_name** – bucket name
- **description** – bucket description
- **org\_id** – org\_id
- **bucket\_name** – bucket name
- **retention\_rules** – retention rules array or single BucketRetentionRules

**Returns** Bucket If the method is called asynchronously, returns the request thread.

```
delete_bucket (bucket)
Delete a bucket # noqa: E501
```

**Parameters** **bucket** – bucket id or Bucket

**Returns** Bucket If the method is called asynchronously, returns the request thread.

```
find_bucket_by_id (id)
Find bucket by ID
```

**Parameters** **id** –

**Returns**



**find\_bucket\_by\_name** (*bucket\_name*)

Find bucket by name

**Parameters** **bucket\_name** – bucket name

**Returns** Bucket

**find\_buckets** ()

Gets all buckets

```
class influxdb_client.domain.Bucket (links=None, id=None, type='user', name=None,  
                                     description=None, org_id=None, rp=None,  
                                     created_at=None, updated_at=None, reten-  
                                     tion_rules=None, labels=None)
```

NOTE: This class is auto generated by OpenAPI Generator. Ref: <https://openapi-generator.tech>

Do not edit the class manually.

Bucket - a model defined in OpenAPI

**created\_at**

Gets the created\_at of this Bucket. # noqa: E501

**Returns** The created\_at of this Bucket. # noqa: E501

**Return type** datetime

**description**

Gets the description of this Bucket. # noqa: E501

**Returns** The description of this Bucket. # noqa: E501

**Return type** str

**id**

Gets the id of this Bucket. # noqa: E501

**Returns** The id of this Bucket. # noqa: E501

**Return type** str

**labels**

Gets the labels of this Bucket. # noqa: E501

**Returns** The labels of this Bucket. # noqa: E501

**Return type** list[Label]

**links**

Gets the links of this Bucket. # noqa: E501

**Returns** The links of this Bucket. # noqa: E501

**Return type** BucketLinks

**name**

Gets the name of this Bucket. # noqa: E501

**Returns** The name of this Bucket. # noqa: E501

**Return type** str

**org\_id**

Gets the org\_id of this Bucket. # noqa: E501

**Returns** The org\_id of this Bucket. # noqa: E501

**Return type** str

**retention\_rules**

Gets the retention\_rules of this Bucket. # noqa: E501

Rules to expire or retain data. No rules means data never expires. # noqa: E501

**Returns** The retention\_rules of this Bucket. # noqa: E501

**Return type** `list[BucketRetentionRules]`

**rp**

Gets the rp of this Bucket. # noqa: E501

**Returns** The rp of this Bucket. # noqa: E501

**Return type** `str`

**to\_dict()**

Returns the model properties as a dict

**to\_str()**

Returns the string representation of the model

**type**

Gets the type of this Bucket. # noqa: E501

**Returns** The type of this Bucket. # noqa: E501

**Return type** `str`

**updated\_at**

Gets the updated\_at of this Bucket. # noqa: E501

**Returns** The updated\_at of this Bucket. # noqa: E501

**Return type** `datetime`

## 2.5 LabelsApi

**class** `influxdb_client.LabelsApi` (`influxdb_client`)

The client of the InfluxDB 2.0 that implements Labels HTTP API endpoint.

**clone\_label** (`cloned_name: str, label: influxdb_client.domain.label.Label`) → `influxdb_client.domain.label.Label`

Creates the new instance of the label as a copy existing label.

**Parameters**

- **cloned\_name** – new label name
- **label** – existing label

**Returns** cloned Label

**create\_label** (`name: str, org_id: str, properties: Dict[str, str] = None`) → `influxdb_client.domain.label.Label`

Creates a new label.

**Parameters**

- **name** – label name
- **org\_id** – organization id
- **properties** – optional label properties

**Returns** created label

**delete\_label** (*label: Union[str, influxdb\_client.domain.label.Label]*)  
Deletes the label.

**Parameters** *label* – label id or Label

**find\_label\_by\_id** (*label\_id: str*)  
Retrieves the label by id.

**Parameters** *label\_id* –

**Returns** Label

**find\_label\_by\_org** (*org\_id*) → List[influxdb\_client.domain.label.Label]  
Gets the list of all labels for given organization.

**Parameters** *org\_id* – organization id

**Returns** list of labels

**find\_labels** () → List[influxdb\_client.domain.label.Label]  
Gets all available labels.

**Returns** labels

**update\_label** (*label: influxdb\_client.domain.label.Label*)  
Updates an existing label name and properties.

**Parameters** *label* – label

**Returns** the updated label

## 2.6 OrganizationsApi

**class** influxdb\_client.OrganizationsApi (*influxdb\_client*)

The client of the InfluxDB 2.0 that implements Organizations HTTP API endpoint.

**class** influxdb\_client.domain.Organization (*links=None, id=None, name=None, description=None, created\_at=None, updated\_at=None, status='active'*)

NOTE: This class is auto generated by OpenAPI Generator. Ref: <https://openapi-generator.tech>

Do not edit the class manually.

Organization - a model defined in OpenAPI

**created\_at**

Gets the created\_at of this Organization. # noqa: E501

**Returns** The created\_at of this Organization. # noqa: E501

**Return type** datetime

**description**

Gets the description of this Organization. # noqa: E501

**Returns** The description of this Organization. # noqa: E501

**Return type** str

**id**

Gets the id of this Organization. # noqa: E501

**Returns** The id of this Organization. # noqa: E501

**Return type** `str`

**links**

Gets the links of this Organization. # noqa: E501

**Returns** The links of this Organization. # noqa: E501

**Return type** `OrganizationLinks`

**name**

Gets the name of this Organization. # noqa: E501

**Returns** The name of this Organization. # noqa: E501

**Return type** `str`

**status**

Gets the status of this Organization. # noqa: E501

If inactive the organization is inactive. # noqa: E501

**Returns** The status of this Organization. # noqa: E501

**Return type** `str`

**to\_dict()**

Returns the model properties as a dict

**to\_str()**

Returns the string representation of the model

**updated\_at**

Gets the updated\_at of this Organization. # noqa: E501

**Returns** The updated\_at of this Organization. # noqa: E501

**Return type** `datetime`

## 2.7 UsersApi

```
class influxdb_client.UsersApi (influxdb_client)
```

```
class influxdb_client.domain.User (id=None, oauth_id=None, name=None, status='active',  
                                   links=None)
```

NOTE: This class is auto generated by OpenAPI Generator. Ref: <https://openapi-generator.tech>

Do not edit the class manually.

User - a model defined in OpenAPI

**id**

Gets the id of this User. # noqa: E501

**Returns** The id of this User. # noqa: E501

**Return type** `str`

**links**

Gets the links of this User. # noqa: E501

**Returns** The links of this User. # noqa: E501

**Return type** `UserLinks`

**name**  
Gets the name of this User. # noqa: E501  
**Returns** The name of this User. # noqa: E501  
**Return type** `str`

**oauth\_id**  
Gets the oauth\_id of this User. # noqa: E501  
**Returns** The oauth\_id of this User. # noqa: E501  
**Return type** `str`

**status**  
Gets the status of this User. # noqa: E501  
If inactive the user is inactive. # noqa: E501  
**Returns** The status of this User. # noqa: E501  
**Return type** `str`

**to\_dict()**  
Returns the model properties as a dict

**to\_str()**  
Returns the string representation of the model

## 2.8 TasksApi

**class** `influxdb_client.TasksApi` (`influxdb_client`)

**cancel\_run** (`task_id: str, run_id: str`)  
Cancels a currently running run. :param task\_id: :param run\_id:

**find\_tasks** (`**kwargs`)  
List tasks.

### Parameters

- **name** (`str`) – only returns tasks with the specified name
- **after** (`str`) – returns tasks after specified ID
- **user** (`str`) – filter tasks to a specific user ID
- **org** (`str`) – filter tasks to a specific organization name
- **org\_id** (`str`) – filter tasks to a specific organization ID
- **limit** (`int`) – the number of tasks to return

### Returns

**get\_logs** (`task_id: str`) → `List[influxdb_client.domain.log_event.LogEvent]`  
Retrieve all logs for a task. :param task\_id: task id

**get\_run** (`task_id: str, run_id: str`) → `influxdb_client.domain.run.Run`  
Get run record for specific task and run id :param task\_id: task id :param run\_id: run id :return: Run for specified task and run id

**get\_runs** (*task\_id*, *\*\*kwargs*) → List[influxdb\_client.domain.run.Run]

Retrieve list of run records for a task

**Parameters**

- **task\_id** – task id
- **after** (*str*) – returns runs after specified ID
- **limit** (*int*) – the number of runs to return
- **after\_time** (*datetime*) – filter runs to those scheduled after this time, RFC3339
- **before\_time** (*datetime*) – filter runs to those scheduled before this time, RFC3339

**retry\_run** (*task\_id: str*, *run\_id: str*)

Retry a task run. :param task\_id: task id :param run\_id: run id

**run\_manually** (*task\_id: str*, *scheduled\_for: <module 'datetime' from 'home/docs/.pyenv/versions/3.6.8/lib/python3.6/datetime.py'> = None*)

Manually start a run of the task now overriding the current schedule.

**Parameters**

- **task\_id** –
- **scheduled\_for** – planned execution

```
class influxdb_client.domain.Task (id=None, type=None, org_id=None, org=None,
                                   name=None, description=None, status=None, labels=None,
                                   authorization_id=None, flux=None, every=None, cron=None,
                                   offset=None, latest_completed=None, created_at=None,
                                   updated_at=None, links=None)
```

NOTE: This class is auto generated by OpenAPI Generator. Ref: <https://openapi-generator.tech>

Do not edit the class manually.

Task - a model defined in OpenAPI

**authorization\_id**

Gets the authorization\_id of this Task. # noqa: E501

The ID of the authorization used when this task communicates with the query engine. # noqa: E501

**Returns** The authorization\_id of this Task. # noqa: E501

**Return type** *str*

**created\_at**

Gets the created\_at of this Task. # noqa: E501

**Returns** The created\_at of this Task. # noqa: E501

**Return type** *datetime*

**cron**

Gets the cron of this Task. # noqa: E501

A task repetition schedule in the form ‘\* \* \* \* \*’; parsed from Flux. # noqa: E501

**Returns** The cron of this Task. # noqa: E501

**Return type** *str*

**description**

Gets the description of this Task. # noqa: E501

An optional description of the task. # noqa: E501

**Returns** The description of this Task. # noqa: E501

**Return type** `str`

**every**

Gets the every of this Task. # noqa: E501

A simple task repetition schedule; parsed from Flux. # noqa: E501

**Returns** The every of this Task. # noqa: E501

**Return type** `str`

**flux**

Gets the flux of this Task. # noqa: E501

The Flux script to run for this task. # noqa: E501

**Returns** The flux of this Task. # noqa: E501

**Return type** `str`

**id**

Gets the id of this Task. # noqa: E501

**Returns** The id of this Task. # noqa: E501

**Return type** `str`

**labels**

Gets the labels of this Task. # noqa: E501

**Returns** The labels of this Task. # noqa: E501

**Return type** `list[Label]`

**latest\_completed**

Gets the latest\_completed of this Task. # noqa: E501

Timestamp of latest scheduled, completed run, RFC3339. # noqa: E501

**Returns** The latest\_completed of this Task. # noqa: E501

**Return type** `datetime`

**links**

Gets the links of this Task. # noqa: E501

**Returns** The links of this Task. # noqa: E501

**Return type** `TaskLinks`

**name**

Gets the name of this Task. # noqa: E501

The name of the task. # noqa: E501

**Returns** The name of this Task. # noqa: E501

**Return type** `str`

**offset**

Gets the offset of this Task. # noqa: E501

Duration to delay after the schedule, before executing the task; parsed from flux, if set to zero it will remove this option and use 0 as the default. # noqa: E501

**Returns** The offset of this Task. # noqa: E501

**Return type** `str`

**org**

Gets the org of this Task. # noqa: E501

The name of the organization that owns this Task. # noqa: E501

**Returns** The org of this Task. # noqa: E501

**Return type** `str`

**org\_id**

Gets the org\_id of this Task. # noqa: E501

The ID of the organization that owns this Task. # noqa: E501

**Returns** The org\_id of this Task. # noqa: E501

**Return type** `str`

**status**

Gets the status of this Task. # noqa: E501

**Returns** The status of this Task. # noqa: E501

**Return type** `TaskStatusType`

**to\_dict()**

Returns the model properties as a dict

**to\_str()**

Returns the string representation of the model

**type**

Gets the type of this Task. # noqa: E501

The type of task, this can be used for filtering tasks on list actions. # noqa: E501

**Returns** The type of this Task. # noqa: E501

**Return type** `str`

**updated\_at**

Gets the updated\_at of this Task. # noqa: E501

**Returns** The updated\_at of this Task. # noqa: E501

**Return type** `datetime`



InfluxDB 2.0 python client library.

**Note:** This library is for use with InfluxDB 2.x. For connecting to InfluxDB 1.x instances, please use the [influxdb-python](#).



---

### InfluxDB 2.0 client features

---

- **Querying data**
  - using the Flux language
  - into csv, raw data, `flux_table` structure
  - *How to queries*
- **Writing data using**
  - Line Protocol
  - Data Point
  - RxPY Observable
  - **Not implemented yet**
    - \* write user types using decorator
    - \* write Pandas DataFrame
  - *How to writes*
- **InfluxDB 2.0 API client for management**
  - the client is generated from the `swagger` by using the `openapi-generator`
  - organizations & users management
  - buckets management
  - tasks management
  - authorizations
  - health check
  - ...
- **Examples**
  - **‘Connect to InfluxDB Cloud’\_**

- **‘How to efficiently import large dataset’\_**
- **‘Efficiency write data from IOT sensor’\_**

## CHAPTER 4

---

### Installation

---

InfluxDB python library uses [RxPY](#) - The Reactive Extensions for Python (RxPY).

**Python 3.6** or later is required.

#### 4.1 pip install

The python package is hosted on Github, you can install latest version directly:

```
pip install influxdb-client
```

Then import the package:

```
import influxdb_client
```

#### 4.2 Setuptools

Install via [Setuptools](#).

```
python setup.py install --user
```

(or `sudo python setup.py install` to install the package for all users)



## CHAPTER 5

---

### Getting Started

---

Please follow the *Installation* and then run the following:

```
from influxdb_client import InfluxDBClient, Point
from influxdb_client.client.write_api import SYNCHRONOUS

bucket = "my-bucket"

client = InfluxDBClient(url="http://localhost:9999", token="my-token", org="my-org")

write_api = client.write_api(write_options=SYNCHRONOUS)
query_api = client.query_api()

p = Point("my_measurement").tag("location", "Prague").field("temperature", 25.3)

write_api.write(bucket=bucket, org="my-org", record=p)

## using Table structure
tables = query_api.query('from(bucket:"my-bucket") |> range(start: -10m)')

for table in tables:
    print(table)
    for row in table.records:
        print(row.values)

## using csv library
csv_result = query_api.query_csv('from(bucket:"my-bucket") |> range(start: -10m)')
val_count = 0
for row in csv_result:
    for cell in row:
        val_count += 1
```





## CHAPTER 6

---

### Indices and tables

---

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



## A

authorization\_id (*influxdb\_client.domain.Task attribute*), 18  
 authorizations\_api() (*influxdb\_client.InfluxDBClient method*), 10

## B

Bucket (*class in influxdb\_client.domain*), 13  
 buckets\_api() (*influxdb\_client.InfluxDBClient method*), 10  
 BucketsApi (*class in influxdb\_client*), 12

## C

cancel\_run() (*influxdb\_client.TasksApi method*), 17  
 clone\_label() (*influxdb\_client.LabelsApi method*), 14  
 close() (*influxdb\_client.InfluxDBClient method*), 10  
 create\_bucket() (*influxdb\_client.BucketsApi method*), 12  
 create\_label() (*influxdb\_client.LabelsApi method*), 14  
 created\_at (*influxdb\_client.domain.Bucket attribute*), 13  
 created\_at (*influxdb\_client.domain.Organization attribute*), 15  
 created\_at (*influxdb\_client.domain.Task attribute*), 18  
 cron (*influxdb\_client.domain.Task attribute*), 18

## D

delete\_api() (*influxdb\_client.InfluxDBClient method*), 10  
 delete\_bucket() (*influxdb\_client.BucketsApi method*), 12  
 delete\_label() (*influxdb\_client.LabelsApi method*), 15  
 description (*influxdb\_client.domain.Bucket attribute*), 13

description (*influxdb\_client.domain.Organization attribute*), 15  
 description (*influxdb\_client.domain.Task attribute*), 18

## E

every (*influxdb\_client.domain.Task attribute*), 19

## F

find\_bucket\_by\_id() (*influxdb\_client.BucketsApi method*), 12  
 find\_bucket\_by\_name() (*influxdb\_client.BucketsApi method*), 12  
 find\_buckets() (*influxdb\_client.BucketsApi method*), 13  
 find\_label\_by\_id() (*influxdb\_client.LabelsApi method*), 15  
 find\_label\_by\_org() (*influxdb\_client.LabelsApi method*), 15  
 find\_labels() (*influxdb\_client.LabelsApi method*), 15  
 find\_tasks() (*influxdb\_client.TasksApi method*), 17  
 flux (*influxdb\_client.domain.Task attribute*), 19

## G

get\_logs() (*influxdb\_client.TasksApi method*), 17  
 get\_run() (*influxdb\_client.TasksApi method*), 17  
 get\_runs() (*influxdb\_client.TasksApi method*), 17

## H

health() (*influxdb\_client.InfluxDBClient method*), 10

## I

id (*influxdb\_client.domain.Bucket attribute*), 13  
 id (*influxdb\_client.domain.Organization attribute*), 15  
 id (*influxdb\_client.domain.Task attribute*), 19  
 id (*influxdb\_client.domain.User attribute*), 16  
 InfluxDBClient (*class in influxdb\_client*), 9

## L

labels (*influxdb\_client.domain.Bucket attribute*), 13  
labels (*influxdb\_client.domain.Task attribute*), 19  
labels\_api() (*influxdb\_client.InfluxDBClient method*), 10  
LabelsApi (*class in influxdb\_client*), 14  
latest\_completed (*influxdb\_client.domain.Task attribute*), 19  
links (*influxdb\_client.domain.Bucket attribute*), 13  
links (*influxdb\_client.domain.Organization attribute*), 16  
links (*influxdb\_client.domain.Task attribute*), 19  
links (*influxdb\_client.domain.User attribute*), 16

## N

name (*influxdb\_client.domain.Bucket attribute*), 13  
name (*influxdb\_client.domain.Organization attribute*), 16  
name (*influxdb\_client.domain.Task attribute*), 19  
name (*influxdb\_client.domain.User attribute*), 16

## O

oauth\_id (*influxdb\_client.domain.User attribute*), 17  
offset (*influxdb\_client.domain.Task attribute*), 19  
org (*influxdb\_client.domain.Task attribute*), 20  
org\_id (*influxdb\_client.domain.Bucket attribute*), 13  
org\_id (*influxdb\_client.domain.Task attribute*), 20  
Organization (*class in influxdb\_client.domain*), 15  
organizations\_api() (*influxdb\_client.InfluxDBClient method*), 10  
OrganizationsApi (*class in influxdb\_client*), 15

## Q

query() (*influxdb\_client.QueryApi method*), 11  
query\_api() (*influxdb\_client.InfluxDBClient method*), 10  
query\_csv() (*influxdb\_client.QueryApi method*), 11  
query\_raw() (*influxdb\_client.QueryApi method*), 11  
query\_stream() (*influxdb\_client.QueryApi method*), 11  
QueryApi (*class in influxdb\_client*), 11

## R

ready() (*influxdb\_client.InfluxDBClient method*), 10  
retention\_rules (*influxdb\_client.domain.Bucket attribute*), 14  
retry\_run() (*influxdb\_client.TasksApi method*), 18  
rp (*influxdb\_client.domain.Bucket attribute*), 14  
run\_manually() (*influxdb\_client.TasksApi method*), 18

## S

status (*influxdb\_client.domain.Organization attribute*), 16

status (*influxdb\_client.domain.Task attribute*), 20  
status (*influxdb\_client.domain.User attribute*), 17

## T

Task (*class in influxdb\_client.domain*), 18  
tasks\_api() (*influxdb\_client.InfluxDBClient method*), 10  
TasksApi (*class in influxdb\_client*), 17  
to\_dict() (*influxdb\_client.domain.Bucket method*), 14  
to\_dict() (*influxdb\_client.domain.Organization method*), 16  
to\_dict() (*influxdb\_client.domain.Task method*), 20  
to\_dict() (*influxdb\_client.domain.User method*), 17  
to\_str() (*influxdb\_client.domain.Bucket method*), 14  
to\_str() (*influxdb\_client.domain.Organization method*), 16  
to\_str() (*influxdb\_client.domain.Task method*), 20  
to\_str() (*influxdb\_client.domain.User method*), 17  
type (*influxdb\_client.domain.Bucket attribute*), 14  
type (*influxdb\_client.domain.Task attribute*), 20

## U

update\_label() (*influxdb\_client.LabelsApi method*), 15  
updated\_at (*influxdb\_client.domain.Bucket attribute*), 14  
updated\_at (*influxdb\_client.domain.Organization attribute*), 16  
updated\_at (*influxdb\_client.domain.Task attribute*), 20  
User (*class in influxdb\_client.domain*), 16  
users\_api() (*influxdb\_client.InfluxDBClient method*), 10  
UsersApi (*class in influxdb\_client*), 16

## W

write() (*influxdb\_client.WriteApi method*), 12  
write\_api() (*influxdb\_client.InfluxDBClient method*), 10  
WriteApi (*class in influxdb\_client*), 12