

---

# **gracedb-sdk Documentation**

***Release 0.1.6***

**Leo Singer**

**Dec 28, 2020**



---

## Contents

---

<b>1</b>	<b>Quick Start</b>	<b>3</b>
<b>2</b>	<b>API</b>	<b>5</b>
	<b>Python Module Index</b>	<b>9</b>
	<b>Index</b>	<b>11</b>



gracedb-sdk is a REST API client for the LIGO/Virgo Gravitational-Wave Candidate Event Database ([GraceDB](#)). It uses the powerful [Requests](#) package for reliable and high-throughput HTTP connection pooling.



# CHAPTER 1

---

## Quick Start

---

Install with `pip`:

```
pip install gracedb-sdk
```





```
class gracedb_sdk.Client (url='https://gracedb.ligo.org/api/', *args, **kwargs)
```

```
    Bases: gracedb_sdk.api.API
```

GraceDB client session.

#### Parameters

- **url** (*str*) – GraceDB Client URL.
- **cert** (*str*, *tuple*) – Client-side X.509 certificate. May be either a single filename if the certificate and private key are concatenated together, or a tuple of the filenames for the certificate and private key.
- **username** (*str*) – Username for basic auth.
- **password** (*str*) – Password for basic auth.
- **force\_noauth** (*bool*, *default=False*) – If true, then do not use any authentication at all.
- **fail\_if\_noauth** (*bool*, *default=False*) – If true, then raise an exception if authentication credentials are not provided.
- **cert\_reload** (*bool*, *default=False*) – If true, then automatically reload the client certificate before it expires.
- **cert\_cert\_reload\_timeout** (*int*, *default=300*) – Reload the certificate this many seconds before it expires.

#### Notes

When a new Client instance is created, the following sources of authentication are tried, in order:

1. If the `force_noauth` keyword argument is true, then perform no authentication at all.
2. If the `cert` keyword argument is provided, then use X.509 client certificate authentication.
3. If the `username` and `password` keyword arguments are provided, then use basic auth.

4. Look for a default X.509 client certificate in:
  - a. the environment variables `X509_USER_CERT` and `X509_USER_KEY`
  - b. the environment variable `X509_USER_PROXY`
  - c. the file `/tmp/x509up_uUID`, where `UID` is your numeric user ID
  - d. the files `~/.globus/usercert.pem` and `~/.globus/userkey.pem`
5. Read the `netrc` file<sup>1</sup> located at `~/.netrc`, or at the path stored in the environment variable `NETRC`, and look for a username and password matching the hostname in the URL.
6. If the `fail_if_noauth` keyword argument is true, and no authentication source was found, then raise a `ValueError`.

The following methods are supported for events:

- `client.events.get()`
- `client.events.search(query=query, sort=sort)`
- `client.events.create(filename=filename, filecontents=filecontents, group=group, pipeline=pipeline, search=search, labels=labels, offline=offline)`
- `client.events.update(event_id, filename=filename, filecontents=filecontents)`
- `client.events[event_id].get()`
- `client.events[event_id].files.get()`
- `client.events[event_id].files[filename].get()`
- `client.events[event_id].labels.get()`
- `client.events[event_id].labels.create(label)`
- `client.events[event_id].labels.delete(label)`
- `client.events[event_id].logs.get()`
- `client.events[event_id].logs.create(comment=comment, filename=filename, filecontents=filecontents, tags=tags)`
- `client.events[event_id].logs[N].tags.create(tag)`
- `client.events[event_id].logs[N].tags.delete(tag)`
- `client.events[event_id].voevents.get()`
- `client.events[event_id].voevents.create(voevent_type={}, internal=internal, open_alert=open_alert, hardware_inj=hardware_inj, skymap_type=skymap_type, skymap_filename=skymap_filename, ProbHasNS=ProbHasNS, ProbHasRemnant=ProbHasRemnant, BNS=BNS, NSBH=NSBH, BBH=BBH, Terrestrial=Terrestrial, MassGap=MassGap, coinc_comment=coinc_comment)`

Analogous methods are supported for superevents:

- `client.superevents.get()`
- `client.superevents.search(query=query, sort=sort)`

---

<sup>1</sup> The `.netrc` file. [https://www.gnu.org/software/inetutils/manual/html\\_node/The-\\_002enetrc-file.html](https://www.gnu.org/software/inetutils/manual/html_node/The-_002enetrc-file.html)

- `client.superevents.create(t_start=t_start, t_0=t_0, t_end=t_end, preferred_event=preferred_event, events=events, labels=labels)`
- `client.superevents.update(superevent_id, t_start=t_start, t_0=t_0, t_end=t_end, preferred_event=preferred_event)`
- `client.superevents[superevent_id].add(event_id)`
- `client.superevents[superevent_id].remove(event_id)`
- `client.superevents[superevent_id].is_exposed()`
- `client.superevents[superevent_id].expose()`
- `client.superevents[superevent_id].unexpose()`
- `client.superevents[superevent_id].signoff('ADV'/'H1'/'L1'/'V1', 'OK'/'NO', comment)`
- `client.superevents[superevent_id].get()`
- `client.superevents[superevent_id].files.get()`
- `client.superevents[superevent_id].files[filename].get()`
- `client.superevents[superevent_id].labels.get()`
- `client.superevents[superevent_id].labels.create(label)`
- `client.superevents[superevent_id].labels.delete(label)`
- `client.superevents[superevent_id].logs.get()`
- `client.superevents[superevent_id].logs.create(comment=comment, filename=filename, filecontents=filecontents, tags=tags)`
- `client.superevents[superevent_id].logs[N].tags.create(tag)`
- `client.superevents[superevent_id].logs[N].tags.delete(tag)`
- `client.superevents[superevent_id].voevents.get()`
- `client.superevents[superevent_id].voevents.create(voevent_type={}, internal=internal, open_alert=open_alert, hardware_inj=hardware_inj, skymap_type=skymap_type, skymap_filename=skymap_filename, ProbHasNS=ProbHasNS, ProbHasRemnant=ProbHasRemnant, BNS=BNS, NSBH=NSBH, BBH=BBH, Terrestrial=Terrestrial, MassGap=MassGap, coinc_comment=coinc_comment)`

## References



**g**

gracedb\_sdk, 5



## C

Client (*class in gracedb\_sdk*), 5

## E

environment variable

NETRC, 6

X509\_USER\_CERT, 6

X509\_USER\_KEY, 6

X509\_USER\_PROXY, 6

## G

gracedb\_sdk (*module*), 5

## N

NETRC, 6

## X

X509\_USER\_CERT, 6

X509\_USER\_KEY, 6

X509\_USER\_PROXY, 6