

---

# **SalesforcePy Documentation**

**Aaron Caffrey**

**Feb 01, 2019**



<b>1</b>	<b>Contributors</b>	<b>3</b>
<b>2</b>	<b>Requirements</b>	<b>5</b>
2.1	Get Started with SalesforcePy . . . . .	5
2.2	Usage Examples . . . . .	6
2.3	Advanced Usage . . . . .	11
2.4	FAQs . . . . .	12
2.5	Contribute . . . . .	13
2.6	SalesforcePy Package Reference . . . . .	14
<b>3</b>	<b>Indices and tables</b>	<b>21</b>
	<b>Python Module Index</b>	<b>23</b>



An absurdly simple package for making Salesforce Rest API calls.

by Aaron Caffrey, Colin Cheevers, Jose Garcia, Tania Prince

This package enables you to produce:

1. A Salesforce client that is reusable, minimalistic, and pythonic.
2. Interfaces that are closely knit into the Salesforce Rest API service specification.
3. Gradual support for the Salesforce API extended family (i.e., Chatter, Analytics, Wave, Tooling, Bulk, Metadata, and so on).



# CHAPTER 1

---

## Contributors

---

Thanks goes to the people who have contributed code to this module, see the [GitHub Contributors](#) page.





Python 2 or 3

## 2.1 Get Started with SalesforcePy

### 2.1.1 Install

#### From Git

Install directly from Git using PyPi.

```
pip install git+ssh://git@github.com/forcedotcom/SalesforcePy.git
```

#### From Local Source

1. Download and extract, or clone this repo.
2. cd into the `/SalesforcePy` directory.
3. Run `pip install ..`

### 2.1.2 Create a Client and Log In

Getting started with SalesforcePy is a three-step process:

1. Import SalesforcePy
2. Create a client
3. Perform a login request

```
# Import SalesforcePy
import SalesforcePy as sfdc

# Create an instance of a Salesforce client. Replace the credentials here with valid
↪ ones.
client = sfdc.client(
    username="jsoap@universalcontainers.com",
    password="p@ssword1",
    client_id="XXXXXXXXXXXXXXXXXXXXXXXXXXXXX",
    ↪XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX",
    client_secret="123456789123456789",
    login_url="test.salesforce.com",
    version = "38.0", # optional, defaults to the latest version for your instance.
    timeout = "30", # optional, defines a connect/read timeout value, if not
    ↪specified requests can hang for minutes or more.
)

# Log in
login_results = client.login()
```

In this example, `login_results[0]` is a dict with the response from the Salesforce OAuth resource. The only supported flow at present is username-password. For more information about the response, see [Understanding the Username-Password OAuth Authentication Flow](#).

You can also use the context manager which handles `login()` and `logout()` automatically.

```
client_args = {'username' : 'jsoap@universalcontainers.com',
               'password' : 'p@ssword1',
               'client_id': 'XXXXXXXXXXXXXXXXXXXXXXXXXXXXX',
               ↪XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX',
               'client_secret' : '123456789123456789'
               }
with sfdc.client(**client_args) as client:
    search_result = client.search('FIND {"test"} RETURNING Case(Id)')
```

## 2.2 Usage Examples

### 2.2.1 Query

Once the login call has been performed successfully, the client maintains the session, allowing you to make API calls. Here's how you can perform a query.

```
query_results = client.query('SELECT Id, Name FROM Account LIMIT 1')
```

In the example above `query_results[0]` is a dict with the response. For more information, See [Execute a SOQL Query](#).

### 2.2.2 Query More

While the client `query()` method helps with making requests for a small amount of data, it is not useful if the total size of records exceeds the standard batch size (2000).

In such scenarios, `query_more()` comes handy. As the example shows, you simply provide the query string in the same way you did previously with `query()`:

```
query_results = client.query_more('SELECT Id, Name FROM Account')
```

In this example, `query_results[0]` is a list of dicts, each of which is a single query result (batch). The behaviour of `query_more()` is to consume `nextRecordsUrl` of each query result recursively until it runs out. For more information, see [Retrieving the Remaining SOQL Query Results in Execute a SOQL Query](#).

### 2.2.3 Insert sObjects

Use the `objects` class from the client to perform DML statements and file i/o with Salesforce objects. Here's an example to demonstrate such insert operation.

```
create_result = client.objects(object_type='Account').insert({"Name" : "SalesforcePy  
→"})
```

In this example, `create_result[0]` is a dict with the response. For more information, see [Create a Record](#).

### 2.2.4 Update sObjects

Update works similarly to insert, with the main difference being that `id` is a required kwarg in `objects` so it is clear which record is to be updated.

```
update_result = client.objects(id='0010Y0000055YG7QAM', object_type='Account').update(  
→{"Name" : "SalesforcePy 2"})
```

In the example above `update_result[0]` will be `None`. This is because the HTTP method used under the hood is PATCH, for which the expected success code is 204. The success code can be found in `update_result[1].status`. For more information, see [Update a Record](#).

### 2.2.5 Delete sObjects

```
delete_result = client.objects(id='0010Y0000055YG7QAM', object_type='Account').  
→delete()
```

In the example above `delete_result[0]` will be `None`. This is because the HTTP method used under the hood is DELETE, for which the expected success code is 204. The success code can be found in `delete_result[1].status`. For more information, see [Delete a Record](#).

### 2.2.6 Query SObject Row

If you know the ID of a record, you can easily retrieve the entire row using this method. This may be preferable to `query()` if you wish to get all fields without specifying them.

```
query_result = client.objects(object_type="Account", id="0010Y00000561jcQAA").  
→query()
```

### 2.2.7 Describe SObject

The describe method retrieves the individual metadata at all levels for the specified SObject.

```
describe_result = client.sobjects(object_type='Account').describe()
```

In the example above `describe_result[0]` will be a dict with the response. For more information, see [sObject Describe](#).

### 2.2.8 Describe Global

The describe global method lists the available objects and their metadata for the organization's data. In addition, it provides the organization encoding, as well as the maximum batch size permitted in queries.

```
describe_global_result = client.sobjects().describe_global()
```

In the example above `describe_global_result[0]` will be a dict with the response. For more information, see [Describe Global](#). The If-Modified-Since header cannot be used with this method.

### 2.2.9 Insert File

There are some special objects in Salesforce such as Attachment, Document, and ContentVersion which allow storage of files as blobs. The following is an example of how to insert a file.

```
# Create a file tuple ordered like so: ( filename, body, content_type )
file = ( "SalesforcePy.txt", "Hello world", "text/plain" )

insert_result = client.sobjects(object_type = "Attachment", binary_field="Body").
→insert({
    "Name": "SalesforcePy",
    "ParentId": "0010Y00000561jcQAA",
    "Description": "An excellent package"
}, binary=file )    # Pass your file through using the binary kwarg
```

### 2.2.10 Search

SOSL search statements can be made as follows:

```
search_result = client.search('FIND {SalesforcePy} RETURNING Account(Id, Name) LIMIT 5
→')
```

### 2.2.11 Execute Anonymous

Anonymous Apex can be executed in a Salesforce organisation like so:

```
ea_result = client.execute_anonymous('system.debug(\'Hello world.\');')
Approval Process
Approvals can be retrieved, submitted and approved/rejected

ap_result = client.approvals(requestBody)
```

See documentation for sample request body.

## 2.2.12 Chatter

Create a feed item (chatter post). It returns a 201 status code for a successful request. See Chatter REST api documentation for information on the expected body to create feed items.

```
# create chatter post
client.chatter.feed_item(body)

# create a comment on a chatter post
client.chatter.feed_comment('feed-elementid', body)
```

## 2.2.13 Wave

### Retrieve a data set

Retrieve a wave data set using the `dataset()` function.

```
client.wave.dataset("opportunities")
```

### Perform a query

Perform a SOQL query using the `wave query()` function.

```
query = {
    "query": "\"q = load \"0Fb0N000000XuvBSAS/0Fc0N000001M5BMSA0\";\nq = filter q by
↪ 'Account.Industry' in
[\"Apparel\", \"Banking\", \"Biotechnology\"];\nq = group q by 'Account.Industry';\nq_
↪ = foreach q generate
'Account.Industry' as 'Account.Industry', count() as 'count';\nq = order q by
↪ 'Account.Industry' asc;\nq = limit q
2000;\""
}

client.wave.query(query)
```

## 2.2.14 Bulk API 2.0

As a general rule, supported Bulk API 2.0 calls can be made from `client.jobs.ingest`. The samples here cover specific calls.

### Create a Job

In this example, we create a job to insert accounts.

```
job_resource = {"object": "Account", "operation": "insert", "lineEnding": "CRLF"}
client.jobs.ingest.create(job_resource=job_resource)
```

For more information on the response for this request, see [Create a Job](#).

### Upload Job Data

In this example, we create a job, then upload a csv file using the job ID.

```
job_resource = {"object": "Account", "operation": "insert", "lineEnding": "CRLF"}
create_result = client.jobs.ingest.create(job_resource=job_resource)

with open("/path/to/accounts.csv") as f:
    csv_file = f.read()
    job_id = create_result[0].get("id")
    batches_result = client.jobs.ingest.batches(job_id=job_id, csv_file=csv_file)
```

For more information on the response for this request, see [Upload Job Data](#).

### Update a job state

In this example, we create a job, upload a csv file using its job ID, then update it with a state of UploadComplete.

```
job_resource = {"object": "Account", "operation": "insert", "lineEnding": "CRLF"}
create_result = client.jobs.ingest.create(job_resource=job_resource)
job_id = create_result[0].get("Id")

with open("/path/to/accounts.csv") as f:
    csv_file = f.read()
    batches_result = client.jobs.ingest.batches(job_id=job_id, csv_file=csv_file)

client.jobs.ingest.update(job_id=job_id, state="UploadComplete")
```

For more information on the response for this request, see [Close or Abort a Job](#).

### Delete a job

In this example, we delete a job based on its ID. Assumed in this example that this value is stored in `job_id`.

```
delete_result = client.jobs.ingest.delete(job_id=job_id)
```

For more information on the response for this request, see [Delete a Job](#).

### Get all jobs

In this example, we get a list of all jobs.

```
get_result = client.jobs.ingest.get()
```

For more information on the response for this request, see [Get All Jobs](#).

### Get Job Info

In this example, we get information for a specific job based on its ID. Assumed in this example that this value is stored in `job_id`.

```
get_result = client.jobs.ingest.get(job_id=job_id)
```

For more information on the response for this request, see [Get Job Info](#).

## Get Job Successes

In this example, we get a successes CSV for a given job based on its ID. Assumed in this example that this value is stored in `job_id`.

```
get_result = client.jobs.ingest.get(job_id=job_id, successes=True)
```

For more information on the response for this request, see [Get Job Successful Record Results](#).

## Get Job Failures

In this example, we get a failures CSV for a given job based on its ID. Assumed in this example that this value is stored in `job_id`.

```
get_result = client.jobs.ingest.get(job_id=job_id, failures=True)
```

For more information on the response for this request, see [Get Job Failed Record Results](#).

## Get Job Unprocessed Rows

In this example, we get an unprocessed rows CSV for a given job based on its ID. Assumed in this example that this value is stored in `job_id`.

```
get_result = client.jobs.ingest.get(job_id=job_id, unprocessed=True)
```

For more information on the response for this request, see [Get Job Unprocessed Record Results](#).

## 2.2.15 Logout

Expires the session by revoking the access token. It returns a 200 status code for a successful token revocation.

```
client.logout()
```

## 2.3 Advanced Usage

### 2.3.1 Using Keyword Arguments

Some of the parameters that are optionally defined at the client level can be defined at the function level as well. Function level arguments supersede the client arguments.

For example, you may want to define an overall timeout value of 30 for all requests but specify a higher value for query calls.

```
client = sfdc.client(
    username=username,
    password=password,
    client_id=client_id,
    client_secret=client_secret,
    timeout="30"
)
query_kwarg= {"timeout" : "60"}
client.query("Select Id FROM Account", **query_kwarg)
```

The following parameters support function level overriding:

- proxies
- timeout
- version

## 2.4 FAQs

### 2.4.1 How do I inspect my organisation schema?

1. Log in to [Workbench](#).
2. Go to **Info > Standard and Custom Objects**.
3. In the Object dropdown, choose the object you wish to inspect (eg. Case) then click Select.
4. Expand Fields. You should find what you're looking for here.

### 2.4.2 How do I test a query?

1. Log in to [Workbench](#).
2. Go to *Queries > SOQL Query*.
3. Enter your query or optionally use the form to help you build the query, then click *Query*.

### 2.4.3 Is it possible to debug requests being made by SalesforcePy?

Yes. Here's an example of how to do it, and what to expect.

```
import logging
import SalesforcePy as sfdc

username = "jsoap@universalcontainers.com"
password = "p@ssword1"
client_id = "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX.
↳XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
client_secret = "123456789123456789"

client = sfdc.client(
    username=username,
    password=password,
    client_id=client_id,
    client_secret=client_secret
)
client.debug(level=logging.INFO) # Tell the client to debug at an info level
client.login() # Outputs "POST https://login.salesforce.com/services/oauth2/token"
↳to logs
```

### 2.4.4 Can I specify a proxy to talk to Salesforce Org in the code?

Yes. Here's an example of how to do it.



```
import SalesforcePy as sfdc

username = "jsoap@universalcontainers.com"
password = "p@ssword1"
client_id = "XXXXXXXXXXXXXXXXXXXXXXXXXXXXX.
↪XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
client_secret = "123456789123456789"

client = sfdc.client(
    username=username,
    password=password,
    client_id=client_id,
    client_secret=client_secret,
    proxies={"https": "localhost:8888/example/"}    # `proxies` kwarg takes a dict as ↪
↪required by the `requests` module.
)
```

## 2.5 Contribute

### 2.5.1 Contribute to the Code

#### Git Workflow

1. Fork this repo.
2. Run `git clone -b <yourbranch> <yourfork>`.
3. From within the project root directory run `pip install`.
4. Develop!
5. Cover your code in `/tests`.
6. Create a pull request to the `forcedotcom:developer` branch.

#### How to Format Your Code

We recommend [pep8](#) Style Guide for Python the best you can. We also encourage using [autopep8](#) for automatic formatting.

### 2.5.2 Contribute to the Documentation

The tools used for documentation are:

- **Read The Docs:** An open source platform for automating, building, versioning, and hosting documentation. For more information see their documentation [here](#).
- **Sphinx:** Sphinx simplifies creating intelligent and beautiful documentation for Python projects (or other documents consisting of multiple reStructuredText sources). For more information on installation and usage, see [Getting Started with Sphinx](#).

Documentation file format is reStructuredText. Learn more with [this handy primer](#).

## 2.6 SalesforcePy Package Reference

### 2.6.1 SalesforcePy.chatter module

New in version 1.0.0.

```
class SalesforcePy.chatter.Chatter (client)
    Bases: SalesforcePy.commons.ApiNamespace

    The Chatter namespace class from which all API calls to a Salesforce organisation are made. .. versionadded::
    1.0.0

    feed_comment (*args, **function_kwarg)

    feed_item (*args, **function_kwarg)

class SalesforcePy.chatter.ChatterFeedComment (session_id, instance_url, _id, body,
                                                **kwargs)
    Bases: SalesforcePy.commons.BaseRequest

    Performs a request to '/services/data/v<api_version>/chatter/feed-elements/<id>/capabilities/comments/items'
    .. versionadded:: 1.0.0

class SalesforcePy.chatter.ChatterFeedItem (session_id, instance_url, body, **kwargs)
    Bases: SalesforcePy.commons.BaseRequest

    Performs a request to '/services/data/v<api_version>/chatter/feed-elements' .. versionadded:: 1.0.0
```

### 2.6.2 SalesforcePy.commons module

New in version 1.0.0.

```
class SalesforcePy.commons.ApiNamespace (client)
    Bases: object

    Base class for API namespaces.

    New in version 1.0.0.

class SalesforcePy.commons.BaseRequest (session_id, instance_url, **kwargs)
    Bases: object

    Base class for all request objects, for convenience, new request types should inherit from this

    New in version 1.0.0.

    get_headers ()
        Returns headers dict for the request.

        Returns headers
        Return type dict

    get_request_url ()
        Returns the request URL. (default: 'https://<instance_url><service>')

        Returns request_url
        Return type string

    get_request_vars ()
        Returns the variables required by request() and other functions.
```

**Returns** (headers, logger, request\_object, response, service)

**Return type** (dict, logging.Logger, requests.Request|None, list|dict|None, string)

**request** ()

Makes request to Salesforce and returns serialised response. Catches any exceptions and appends them to *self.exceptions*.

**return** response: Salesforce response, if available

**rtype** list|dict|None

**set\_proxies** (*proxies*)

Sets *proxies* for this class. :param proxies: A dict containing proxies to use (see: <http://docs.python-requests.org/en/master/user/advanced/#proxies>) :type: dict

**class** SalesforcePy.common.OAuthRequest (*session\_id, instance\_url, \*\*kwargs*)

Bases: *SalesforcePy.common.BaseRequest*

Base class for all OAuth request objects

New in version 1.0.0.

**get\_request\_url** ()

Returns the request URL. (default: 'https://<instance\_url><service>')

**Returns** request\_url

**Return type** string

**request** ()

Makes request to Salesforce and returns serialised response. Catches any exceptions and appends them to *self.exceptions*.

**return** response: Salesforce response, if available

**rtype** list|dict|None

**exception** SalesforcePy.common.SFDCRequestException

Bases: *exceptions.Exception*

This exception is raised when we fail to complete requests to the SFDC REST API. .. versionadded:: 1.0.0

SalesforcePy.common.delete\_request (*base\_request*)

Performs DELETE request for the class provided. :param: base\_request: Class with which to make request. :type: BaseRequest :return: response :rtype: requests.Response

SalesforcePy.common.get\_request (*base\_request*)

Performs GET request for the class provided. :param: base\_request: Class with which to make request. :type: BaseRequest :return: response :rtype: requests.Response

SalesforcePy.common.kwarg\_adder (*func*)

Decorator to add the kwargs from the client to the kwargs at the function level. If the same parameters are used in both, the function level kwarg will supersede the one at the client level. :param func: client function to add client kwargs to :return: the function with updated kwargs

SalesforcePy.common.patch\_request (*base\_request*)

Performs PATCH request for the class provided. :param: base\_request: Class with which to make request. :type: BaseRequest :return: response :rtype: requests.Response

SalesforcePy.common.post\_request (*base\_request*)

Performs POST request for the class provided. :param: base\_request: Class with which to make request. :type: BaseRequest :return: response :rtype: requests.Response

`SalesforcePy.common.put_request` (*base\_request*)

Performs PUT request for the class provided. :param: *base\_request*: Class with which to make request. :type: *BaseRequest* :return: response :rtype: *requests.Response*

### 2.6.3 SalesforcePy.jobs module

**class** `SalesforcePy.jobs.Batches` (*session\_id, instance\_url, api\_version, job\_id, csv\_file, \*\*kwargs*)

Bases: *SalesforcePy.common.BaseRequest*

Performs a PUT request to `'/services/data/vX.XX/jobs/ingest/<job_id>/batches'` .. versionadded:: 1.1.0

**class** `SalesforcePy.jobs.CreateJob` (*session\_id, instance\_url, api\_version, request\_body, \*\*kwargs*)

Bases: *SalesforcePy.common.BaseRequest*

Performs a POST request to `'/services/data/vX.XX/jobs/ingest'` .. versionadded:: 1.1.0

**class** `SalesforcePy.jobs.DeleteJob` (*session\_id, instance\_url, api\_version, job\_id, \*\*kwargs*)

Bases: *SalesforcePy.common.BaseRequest*

Performs a DELETE request to `'/services/data/vX.XX/jobs/ingest/<job_id>'` .. versionadded:: 1.1.0

**class** `SalesforcePy.jobs.GetJob` (*session\_id, instance\_url, api\_version, \*\*kwargs*)

Bases: *SalesforcePy.common.BaseRequest*

Performs a GET request to `'/services/data/vX.XX/jobs/ingest'` .. versionadded:: 1.1.0

**class** `SalesforcePy.jobs.Ingest` (*client*)

Bases: *SalesforcePy.common.ApiNamespace*

**batches** (*\*args, \*\*function\_kwarg*)

**create** (*\*args, \*\*function\_kwarg*)

**delete** (*\*args, \*\*function\_kwarg*)

**get** (*\*args, \*\*function\_kwarg*)

**update** (*\*args, \*\*function\_kwarg*)

**class** `SalesforcePy.jobs.Jobs` (*client*)

Bases: *SalesforcePy.common.ApiNamespace*

**class** `SalesforcePy.jobs.UpdateJob` (*session\_id, instance\_url, api\_version, job\_id, request\_body, \*\*kwargs*)

Bases: *SalesforcePy.common.BaseRequest*

Performs a PATCH request to `'/services/data/vX.XX/jobs/ingest/<job_id>'` .. versionadded:: 1.1.0

### 2.6.4 SalesforcePy.sfdc module

**class** `SalesforcePy.sfdc.ApprovalProcess` (*session\_id, instance\_url, \*\*kwargs*)

Bases: *SalesforcePy.common.BaseRequest*

**Returns a list of all approval processes. Can also be used to submit a particular record if that entity supports an approval process and one has already been defined. It also supports specifying a collection of different Process Approvals requests to have them all executed in bulk.**

New in version 1.0.0.

---

```

class SalesforcePy.sfdc.Client (*args, **kwargs)
    Bases: object

    The client class from which all API calls to a Salesforce organisation are made. .. versionadded:: 1.0.0

    approvals (*args, **function_kwarg)

    debug (*args, **function_kwarg)

    execute_anonymous (*args, **function_kwarg)

    login (*args, **function_kwarg)

    logout (*args, **function_kwarg)

    query (*args, **function_kwarg)

    query_more (*args, **function_kwarg)

    search (*args, **function_kwarg)

    set_api_version (*args, **function_kwarg)

    set_instance_url (url)
        Strips the protocol from url and assigns the value to self.instance_url

        Parameters url (string) – Instance URL used to make requests (eg.
            'https://eu11.salesforce.com')

    subjects (*args, **function_kwarg)

class SalesforcePy.sfdc.ExecuteAnonymous (session_id, instance_url, ab, **kwargs)
    Bases: SalesforcePy.common.BaseRequest

    Performs a request to '/services/data/vX.XX/tooling/executeAnonymous/' .. versionadded:: 1.0.0

class SalesforcePy.sfdc.Login (username, password, client_id, client_secret, **kwargs)
    Bases: SalesforcePy.common.OAuthRequest

    Performs a request to '/services/oauth2/token' .. versionadded:: 1.0.0

    get_payload ()
        Returns the payload dict to be used in the request.

        Returns OAuth2 request body required to obtain access token.

        Return type dict

    get_session_id ()
        Returns the session ID obtained if the login request was successful

        Returns Session ID

        Return type string

    request ()
        Gets the result of super for this method, then assigns the access_token to session_id. Returns request
        response.

        return Response dict

        rtype dict

exception SalesforcePy.sfdc.LoginException
    Bases: exceptions.Exception

    Exception thrown during due to login failure. .. versionadded:: 1.0.0

```

**class** SalesforcePy.sfdc.**Logout** (*session\_id, instance\_url, \*\*kwargs*)

Bases: *SalesforcePy.commonsoAuthRequest*

Performs a request to *'/services/oauth2/revoke'* .. versionadded:: 1.0.0

**class** SalesforcePy.sfdc.**Query** (*session\_id, instance\_url, query\_string, \*\*kwargs*)

Bases: *SalesforcePy.commonsoBaseRequest*

Performs a request to *'/services/data/vX.XX/query'* .. versionadded:: 1.0.0

**class** SalesforcePy.sfdc.**QueryMore** (*session\_id, instance\_url, query\_string, \*\*kwargs*)

Bases: *SalesforcePy.commonsoBaseRequest*

Performs recursive requests to *'/services/data/vX.XX/query'* when there are multiple batches to process. .. versionadded:: 1.0.0

**request** (*\*args*)

Makes a *Query* request for the initial query string, then calls itself recursively to request all remaining batches, if there are any. This method will break the recursion and return all when the last batch processed contains a *done* value equal to *True*.

**param** results: All queried batch results obtained in prior recursions of this method.

**type** results: [dict]

**return** A list of dicts where each dict is a batch of query results

**rtype** [dict]

**class** SalesforcePy.sfdc.**SObjectBlob** (*\_client, service, http\_method*)

Bases: *SalesforcePy.commonsoBaseRequest*

Perform a request to *'/services/data/vX.XX/objects'* where file i/o is necessary. .. versionadded:: 1.0.0

**request** ()

Returns the request response.

**Returns** response

**Return type** dict

**set\_request\_body** (*\*\*kwargs*)

Creates binary request body by merging *entity, json\_body, file\_content\_type, field, filename* and *content* from *\*\*kwargs* into a binary body template. Sets *request\_body* instance variable with the result.

Note: *content* can be either a *file* or a raw value.

**param** *\*\*kwargs*:

**type** *\*\*kwargs*: string

**class** SalesforcePy.sfdc.**SObjectController** (*\_client, object\_type, \_id, binary\_field, api\_version, external\_id*)

Bases: object

A special class that controls insert/update/delete/query/describe of SObject resources. .. versionadded:: 1.0.0

**delete** (*\*args, \*\*function\_kwarg*)

**describe** (*\*args, \*\*function\_kwarg*)

**describe\_global** (*\*args, \*\*function\_kwarg*)

**get\_service** ()

Returns the correct object service depending on whether the controller contains an *id* instance variable

**Returns** service

**Return type** string

**insert** (\*args, \*\*function\_kwarg)

**query** (\*args, \*\*function\_kwarg)

**update** (\*args, \*\*function\_kwarg)

**upsert** (\*args, \*\*function\_kwarg)

**class** SalesforcePy.sfdc.SObjects(\_client, \*\*kwargs)

Bases: *SalesforcePy.common.BaseRequest*

Perform a request to '/services/data/vX.XX/objects' .. versionadded:: 1.0.0

**request** ()

Makes the appropriate request depending on the *http\_method*. Supported now are: 'GET', 'POST', 'PATCH', and 'DELETE'. Returns request response.

Note: As successful 'PATCH' and 'DELETE' responses return *NO CONTENT*, this method will return *None*. It may be advisable to check the *status* of the *SObject* instance returned as an additional factor in determining whether the request succeeded.

**return** response dict

**rtype** dict|None

**class** SalesforcePy.sfdc.Search(session\_id, instance\_url, search\_string, \*\*kwargs)

Bases: *SalesforcePy.common.BaseRequest*

Performs a request to '/services/data/vX.XX/search/' .. versionadded:: 1.0.0

SalesforcePy.sfdc.client(username, password, client\_id, client\_secret, \*\*kwargs)

Builds a *Client* and returns it.

New in version 1.0.0.

Note: if any of the required parameters are missing, a *ValueError* will be raised.

#### Parameters

- *\*username (string)* - Salesforce username.
- *\*password (string)* - Salesforce password.
- *\*client\_id (string)* - Salesforce client ID.
- *\*client\_secret (string)* - Salesforce client secret.
- *\*\*kwargs* - kwargs (see below)

#### Keyword Arguments

- *login\_url (string)* – Salesforce login URL without protocol Default: 'login.salesforce.com'
- *protocol (string)* – Protocol (future use)
- *proxies (dict)* –  

A dict containing proxies to be used by *requests* module. Ex: {"https": "example.org:443"}

Default: *None*
- *timeout ('string')* – Tell Requests to stop waiting for a response after a given number of seconds

**returns** client  
**rtype** Client  
**raises** ValueError

### 2.6.5 SalesforcePy.wave module

New in version 1.0.0.

**class** SalesforcePy.wave.**Wave**(*client*)

Bases: *SalesforcePy.common.ApiNamespace*

The Wave namespace class from which all API calls to a Salesforce organisation are made. .. versionadded:: 1.0.0

**dataset** (\*args, \*\*function\_kwarg)

**query** (\*args, \*\*function\_kwarg)

**class** SalesforcePy.wave.**WaveDataSet**(*session\_id, instance\_url, api\_name, \*\*kwargs*)

Bases: *SalesforcePy.common.BaseRequest*

Performs a request to `'/services/data/<api_version>/wave/datasets/<api_name>'` .. versionadded:: 1.0.0

**class** SalesforcePy.wave.**WaveQuery**(*session\_id, instance\_url, query, \*\*kwargs*)

Bases: *SalesforcePy.common.BaseRequest*

Performs a request to `'/services/data/<api_version>/wave/query'` .. versionadded:: 1.0.0



## CHAPTER 3

---

### Indices and tables

---

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



### C

`chatter`, [14](#)  
`commons`, [14](#)

### j

`jobs`, [16](#)

### S

`SalesforcePy.chatter`, [14](#)  
`SalesforcePy.commons`, [14](#)  
`SalesforcePy.jobs`, [16](#)  
`SalesforcePy.sfdc`, [16](#)  
`SalesforcePy.wave`, [20](#)  
`sfdc`, [16](#)

### W

`wave`, [20](#)



## A

ApiNamespace (class in SalesforcePy.common), 14  
ApprovalProcess (class in SalesforcePy.sfdc), 16  
approvals() (SalesforcePy.sfdc.Client method), 17

## B

BaseRequest (class in SalesforcePy.common), 14  
Batches (class in SalesforcePy.jobs), 16  
batches() (SalesforcePy.jobs.Ingest method), 16

## C

Chatter (class in SalesforcePy.chatter), 14  
chatter (module), 14  
ChatterFeedComment (class in SalesforcePy.chatter), 14  
ChatterFeedItem (class in SalesforcePy.chatter), 14  
Client (class in SalesforcePy.sfdc), 16  
client() (in module SalesforcePy.sfdc), 19  
common (module), 14  
create() (SalesforcePy.jobs.Ingest method), 16  
CreateJob (class in SalesforcePy.jobs), 16

## D

dataset() (SalesforcePy.wave.Wave method), 20  
debug() (SalesforcePy.sfdc.Client method), 17  
delete() (SalesforcePy.jobs.Ingest method), 16  
delete() (SalesforcePy.sfdc.SObjectController method), 18  
delete\_request() (in module SalesforcePy.common), 15  
DeleteJob (class in SalesforcePy.jobs), 16  
describe() (SalesforcePy.sfdc.SObjectController method), 18  
describe\_global() (SalesforcePy.sfdc.SObjectController method), 18

## E

execute\_anonymous() (SalesforcePy.sfdc.Client method), 17  
ExecuteAnonymous (class in SalesforcePy.sfdc), 17

## F

feed\_comment() (SalesforcePy.chatter.Chatter method), 14  
feed\_item() (SalesforcePy.chatter.Chatter method), 14

## G

get() (SalesforcePy.jobs.Ingest method), 16  
get\_headers() (SalesforcePy.common.BaseRequest method), 14  
get\_payload() (SalesforcePy.sfdc.Login method), 17  
get\_request() (in module SalesforcePy.common), 15  
get\_request\_url() (SalesforcePy.common.BaseRequest method), 14  
get\_request\_url() (SalesforcePy.common.OAuthRequest method), 15  
get\_request\_vars() (SalesforcePy.common.BaseRequest method), 14  
get\_service() (SalesforcePy.sfdc.SObjectController method), 18  
get\_session\_id() (SalesforcePy.sfdc.Login method), 17  
GetJob (class in SalesforcePy.jobs), 16

## I

Ingest (class in SalesforcePy.jobs), 16  
insert() (SalesforcePy.sfdc.SObjectController method), 19

## J

Jobs (class in SalesforcePy.jobs), 16  
jobs (module), 16

## K

kwargs\_adder() (in module SalesforcePy.common), 15

## L

Login (class in SalesforcePy.sfdc), 17  
login() (SalesforcePy.sfdc.Client method), 17  
LoginException, 17  
Logout (class in SalesforcePy.sfdc), 17

logout() (SalesforcePy.sfdc.Client method), 17

## O

OAuthRequest (class in SalesforcePy.common), 15

## P

patch\_request() (in module SalesforcePy.common), 15

post\_request() (in module SalesforcePy.common), 15

put\_request() (in module SalesforcePy.common), 15

## Q

Query (class in SalesforcePy.sfdc), 18

query() (SalesforcePy.sfdc.Client method), 17

query() (SalesforcePy.sfdc.SObjectController method), 19

query() (SalesforcePy.wave.Wave method), 20

query\_more() (SalesforcePy.sfdc.Client method), 17

QueryMore (class in SalesforcePy.sfdc), 18

## R

request() (SalesforcePy.common.BaseRequest method), 15

request() (SalesforcePy.common.OAuthRequest method), 15

request() (SalesforcePy.sfdc.Login method), 17

request() (SalesforcePy.sfdc.QueryMore method), 18

request() (SalesforcePy.sfdc.SObjectBlob method), 18

request() (SalesforcePy.sfdc.SObjects method), 19

## S

SalesforcePy.chatter (module), 14

SalesforcePy.common (module), 14

SalesforcePy.jobs (module), 16

SalesforcePy.sfdc (module), 16

SalesforcePy.wave (module), 20

Search (class in SalesforcePy.sfdc), 19

search() (SalesforcePy.sfdc.Client method), 17

set\_api\_version() (SalesforcePy.sfdc.Client method), 17

set\_instance\_url() (SalesforcePy.sfdc.Client method), 17

set\_proxies() (SalesforcePy.common.BaseRequest method), 15

set\_request\_body() (SalesforcePy.sfdc.SObjectBlob method), 18

sfdc (module), 16

SFDCRequestException, 15

SObjectBlob (class in SalesforcePy.sfdc), 18

SObjectController (class in SalesforcePy.sfdc), 18

SObjects (class in SalesforcePy.sfdc), 19

sobjects() (SalesforcePy.sfdc.Client method), 17

## U

update() (SalesforcePy.jobs.Ingest method), 16

update() (SalesforcePy.sfdc.SObjectController method), 19

UpdateJob (class in SalesforcePy.jobs), 16

upsert() (SalesforcePy.sfdc.SObjectController method), 19

## W

Wave (class in SalesforcePy.wave), 20

wave (module), 20

WaveDataSet (class in SalesforcePy.wave), 20

WaveQuery (class in SalesforcePy.wave), 20