# STMOP API

The Simple Text Oriented Messaging Protocol

# Stomp.py API

```
>>> import time
>>> import sys
>>>
>>> import stomp
>>>
>>> class MyListener(stomp.ConnectionListener):
>>>     def on_error(self, headers, message):
>>>         print('received an error "%s"' % message)
>>>     def on_message(self, headers, message):
>>>         print('received a message "%s"' % message)

>>> conn = stomp.Connection()
>>> conn.set_listener('', MyListener())
>>> conn.start()
>>> conn.connect('admin', 'password', wait=True)
>>>
>>> conn.subscribe(destination='/queue/test', id=1, ack='auto')
>>>
>>> conn.send(body=' '.join(sys.argv[1:]), destination='/queue/test')
>>>
>>> time.sleep(2)
>>> conn.disconnect()
```

# Establishing a connection

```
>>> import stomp
>>> c = stomp.Connection([('127.0.0.1', 62613)])
>>> c.start()
>>> c.connect('admin', 'password', wait=True)
```

`Protocol12.connect`(*username=None*, *passcode=None*, *wait=False*, *headers=None*, ***keyword_headers*)

Send a STOMP CONNECT frame. Differs from 1.0 and 1.1 versions in that the HOST header is enforced.

**Parameters:**

- **username** (*str*) – optionally specify the login user
- **passcode** (*str*) – optionally specify the user password
- **wait** (*bool*) – wait for the connection to complete before returning
- **headers** (*dict*) – a map of any additional headers to send with the subscription
- **keyword_headers** – any additional headers to send with the subscription

# Sending and receiving messages

`Protocol12.send`(*destination*, *body*, *content_type=None*, *headers=None*, ***keyword_headers*)

Send a message to a destination in the messaging system (as per https://stomp.github.io/stomp-specification-1.2.html#SEND)

**Parameters:**

- **destination** (*str*) – the destination (such as a message queue - for example '/queue/test' - or a message topic)
- **body** – the content of the message
- **content_type** (*str*) – the MIME type of message
- **headers** (*dict*) – additional headers to send in the message frame
- **keyword_headers** – any additional headers the broker requires

`Protocol12.`**`subscribe`**(*destination, id, ack='auto', headers=None, **keyword_headers*)

Subscribe to a destination

**Parameters:**

- **destination** (*str*) – the topic or queue to subscribe to
- **id** (*str*) – the identifier to uniquely identify the subscription
- **ack** (*str*) – either auto, client or client-individual (see https://stomp.github.io/stomp-specification-1.2.html#SUBSCRIBE for more info)
- **headers** (*dict*) – a map of any additional headers to send with the subscription
- **keyword_headers** – any additional headers to send with the subscription

# Transactions

```
>>> conn.subscribe('/queue/test', id=5)
>>> txid = conn.begin()
>>> conn.send('/queue/test', 'test1', transaction=txid)
>>> conn.send('/queue/test', 'test2', transaction=txid)
>>> conn.send('/queue/test', 'test3', transaction=txid)
>>> conn.commit(txid)
```

```
>>> conn.subscribe('/queue/test', id=6)
>>> txid = conn.begin()
>>> conn.send('/queue/test', 'test4', transaction=txid)
>>> conn.send('/queue/test', 'test5', transaction=txid)
>>> conn.abort(txid)
```

# Disconnect

Stomp.py supports graceful shutdown/disconnections through a receipt parameter (automatically generated if you don't provide it). The connection is only dropped when the server sends back a response to that receipt:

```
>>> conn.disconnect()
on_send DISCONNECT {'receipt': '825a5cd6-9e3c-4a72-8051-72348a94f5ce'}
on_receipt {'receipt-id': '825a5cd6-9e3c-4a72-8051-72348a94f5ce'}
on_disconnected
```

- http://jasonrbriggs.github.io/stomp.py/index.html