

---

# Sender Documentation

*Release 0.3*

**Shipeng Feng**

May 11, 2016



<b>1</b>	<b>Installation</b>	<b>3</b>
<b>2</b>	<b>Quickstart</b>	<b>5</b>
<b>3</b>	<b>Message</b>	<b>7</b>
<b>4</b>	<b>Mail</b>	<b>9</b>
<b>5</b>	<b>Attachment</b>	<b>11</b>
<b>6</b>	<b>API</b>	<b>13</b>
<b>7</b>	<b>Changelog</b>	<b>15</b>
7.1	Version 0.1 . . . . .	15
7.2	Version 0.2 . . . . .	15
7.3	Version 0.3 . . . . .	15
<b>8</b>	<b>Contribute</b>	<b>17</b>
	<b>Python Module Index</b>	<b>19</b>



Sender provides a simple interface to set up SMTP and send email messages.



---

## Installation

---

Install with the following command:

```
$ pip install sender
```





---

### Quickstart

---

Sender is really easy to use. Emails are managed through a *Mail* instance:

```
from sender import Mail

mail = Mail()

mail.send_message("Hello", fromaddr="from@example.com",
                  to="to@example.com", body="Hello world!")
```



---

## Message

---

To send one message, we need to create a *Message* instance:

```
from sender import Message

msg = Message("demo subject", fromaddr="from@example.com",
              to="to@example.com")
```

You can also set attribute individually:

```
msg.body = "demo body"
```

It is possible to set *fromaddr* with a two-element tuple:

```
# fromaddr will be "Name <name@example.com>"
msg = Message("Hello", fromaddr=("Name", "name@example.com"))
```

The message could have a plain text body and(or) HTML:

```
msg.body = "hello"
msg.html = "<h1>hello</h1>"
```

Let's construct one full message with all options:

```
msg = Message("msg subject")
msg.fromaddr = ("Admin", "admin@example.com")
msg.to = "to@example.com"
msg.body = "this is a msg plain text body"
msg.html = "<b>Hello</b>"
msg.cc = "cc@example.com"
msg.bcc = ["bcc01@example.com", "bcc02@example.com"]
msg.reply_to = "reply@example.com"
msg.date = time.time()
msg.charset = "utf-8"
msg.extra_headers = {}
msg.mail_options = []
msg.rcpt_options = []
```



---

## Mail

---

To connect to the SMTP server and send messages, we need to create a *Mail* instance:

```
from sender import Mail

mail = Mail("localhost", port=25, username="username", password="pass",
            use_tls=False, use_ssl=False, debug_level=None)
```

You can set `fromaddr` to a mail instance, if the message sent by this mail instance does not set `fromaddr`, this global `fromaddr` will be used:

```
mail.fromaddr = ("Name", "name@example.com")
```

Now let's send our messages:

```
mail.send(msg)
# or an iterable of messages
mail.send([msg1, msg2, msg3])
```

There is one shortcut for sending one message quickly:

```
mail.send_message("hello", to="to@example.com", body="hello body")
```



---

## Attachment

---

It is quite easy to add attachments, we need *Attachment* instance:

```
from sender import Attachment

with open("logo.jpg") as f:
    attachment = Attachment("logo.jpg", "image/jpeg", f.read())

msg.attach(attachment)
```

If you have multiple attachments:

```
msg.attach(attach01)
msg.attach(attach02)
msg.attach(attach03)
# or an iterable of attachments
msg.attach((attach01, attach02, attach03))
```

There is one shortcut for attaching one attachment quickly:

```
msg.attach_attachment("logo.jpg", "image/jpeg", raw_data)
```





---

```
class sender.Mail (host='localhost', username=None, password=None, port=25, use_tls=False,  

                   use_ssl=False, debug_level=None, fromaddr=None)
```

Sender Mail main class. This class is used for manage SMTP server connections and send messages.

#### Parameters

- **host** – smtp server host, default to be 'localhost'
- **username** – smtp server authentication username
- **password** – smtp server authentication password
- **port** – smtp server port, default to be 25
- **use\_tls** – put the SMTP connection in TLS (Transport Layer Security) mode, default to be False
- **use\_ssl** – put the SMTP connection in SSL mode, default to be False
- **debug\_level** – the debug output level
- **fromaddr** – default sender for all messages sent by this mail instance

#### connection

Open one connection to the SMTP server.

```
send (message_or_messages)
```

Sends a single message or multiple messages.

**Parameters** **message\_or\_messages** – one message instance or one iterable of message instances.

```
send_message (*args, **kwargs)
```

Shortcut for send.

```
class sender.Message (subject=None, to=None, body=None, html=None, fromaddr=None, cc=None,  

                     bcc=None, attachments=None, reply_to=None, date=None, charset='utf-8', ex-  

                     tra_headers=None, mail_options=None, rcpt_options=None)
```

One email message.

#### Parameters

- **subject** – message subject
- **to** – message recipient, should be one or a list of addresses
- **body** – plain text content body
- **html** – HTML content body

- **fromaddr** – message sender, can be one address or a two-element tuple
- **cc** – CC list, should be one or a list of addresses
- **bcc** – BCC list, should be one or a list of addresses
- **attachments** – a list of attachment instances
- **reply\_to** – reply-to address
- **date** – message send date, seconds since the Epoch, default to be `time.time()`
- **charset** – message charset, default to be 'utf-8'
- **extra\_headers** – a dictionary of extra headers
- **mail\_options** – a list of ESMTP options used in MAIL FROM commands
- **rcpt\_options** – a list of ESMTP options used in RCPT commands

**attach** (*attachment\_or\_attachments*)

Adds one or a list of attachments to the message.

Parameters **attachment\_or\_attachments** – one or an iterable of attachments

**attach\_attachment** (*\*args, \*\*kwargs*)

Shortcut for attach.

**class** `sender.Attachment` (*filename=None, content\_type=None, data=None, disposition='attachment', headers={}*)

File attachment information.

**Parameters**

- **filename** – filename
- **content\_type** – file mimetype
- **data** – raw data
- **disposition** – content-disposition, default to be 'attachment'
- **headers** – a dictionary of headers, default to be { }

---

## Changelog

---

### 7.1 Version 0.1

First public preview release.

### 7.2 Version 0.2

Released on Sep 18th 2014

- Do message validation on all recipients instead of to

### 7.3 Version 0.3

Released on May 11th 2016

- Added Python 3.x support



---

## Contribute

---

Pull requests are welcomed, thank you for your suggestions!



**S**

sender, [3](#)





## A

`attach()` (`sender.Message` method), [14](#)  
`attach_attachment()` (`sender.Message` method), [14](#)  
`Attachment` (class in `sender`), [14](#)

## C

`connection` (`sender.Mail` attribute), [13](#)

## M

`Mail` (class in `sender`), [13](#)  
`Message` (class in `sender`), [13](#)

## S

`send()` (`sender.Mail` method), [13](#)  
`send_message()` (`sender.Mail` method), [13](#)  
`sender` (module), [1](#)