

---

# **pyOutlook Documentation**

***Release 3.3.1b0***

**Jens Astrup**

**Aug 06, 2017**



---

## Contents

---

<b>1</b>	<b>About:</b>	<b>1</b>
1.1	Python Versions . . . . .	1
<b>2</b>	<b>Recommended:</b>	<b>3</b>
<b>3</b>	<b>Notes:</b>	<b>5</b>
<b>4</b>	<b>Contents:</b>	<b>7</b>
4.1	Installation . . . . .	7
4.2	Quick Start . . . . .	7
4.3	pyOutlook . . . . .	9
<b>5</b>	<b>Indices and tables</b>	<b>17</b>



# CHAPTER 1

---

## About:

---

pyOutlook was created after I found myself attempting to connect to the Outlook REST API in multiple projects. This provided some much needed uniformity. It's easier to deal with than the win32com package by Microsoft, but obviously has a far smaller scope.

## Python Versions

pyOutlook is only tested in, and targets, Python 3.5, 3.6, and 2.7.



## CHAPTER 2

---

### Recommended:

---

pyOutlook does not handle OAuth for the access tokens provided by Outlook. These are provided by you via the OutlookAccount class as a string. There are various OAuth packages out there: (pip install) oauth2, python-oauth2, requests\_oauthlib, etc that can facilitate the process.





## CHAPTER 3

---

Notes:

---

All protected & private methods and anything under `pyOutlook.internal` is subject to change without deprecation warnings



## CHAPTER 4

---

### Contents:

---

## Installation

### Pip

```
pip install pyOutlook
```

### Source

pyOutlook's [PyPI](#) page has a tar.gz and zip distribution for each release.

## Quick Start

### Instantiation

pyOutlook interacts with Outlook messages and folders through the class `OutlookAccount()`. The `OutlookAccount` acts as a gatekeeper to the other methods available, and stores the access token associated with an account.

Instantiation Example:

```
from pyOutlook import OutlookAccount
account_one = OutlookAccount('token 1')
account_two = OutlookAccount('token 2')
```

From here you can access any of the methods as documented in the [pyOutlook](#) section. Here are two examples of accessing an inbox and sending a new email.

## Examples

### Retrieving Emails

Through the OutlookAccount class you can call one of many methods - `get_messages()`, `inbox()`, etc. These methods return a list of *Message* objects, allowing you to access the attributes therein.

```
inbox = account.inbox()
inbox[0].body
>>> 'A very fine body'
```

### Sending Emails

As above, you can send emails through the OutlookAccount class. There are two methods for sending emails - one allows chaining of methods and the other takes all arguments upfront and immediately sends.

### Message

You can create an instance of a *Message* and then send from there.

```
from pyOutlook import *
# or from pyOutlook.core.message import Message

account = OutlookAccount('token')
message = Message(account, 'A body', 'A subject', [Contact('to@email.com')])
message.attach(bytes('some bytes', 'utf-8'), 'bytes.txt')
message.send()
```

### new\_email()

This returns a *Message* instance.

```
body = 'I\'m sending an email through Python. <br> Best, <br>Me'

email = account.new_email(body=body, subject='Hey there', to=Contact('myemail@domain.
↵com'))
email.sender = Contact('some_other_account@email.com')
email.send()
```

Note that HTML formatting is accepted in the message body.

### send\_email()

This method takes all of its arguments at once and then sends.

```
account_one.send_email(
    to=[Contact('myemail@domain.com')],
    # or to=['myemail@domain.com']
    subject='Hey there',
    body="I'm sending an email through Python. <br> Best, <br> Me",
)
```

## Contacts

All recipients, and the sender attribute, in *Messages* are represented by *Contacts*. Right now, this allows you to retrieve the name of a recipient, if provided by Outlook.

```
message = account.inbox()[0]
message.sender.name
>>> 'Dude'
```

When providing recipients to *Message* you can provide them either as a list of strings, or a list of *Contacts*. I prefer the latter, as there are further options in the Outlook API for interacting with Contacts - functionality for those may be added in the future.

## pyOutlook

### Outlook Account

**class** `pyOutlook.core.main.OutlookAccount` (*access\_token*)

Sets up access to Outlook account for all methods & classes.

**access\_token**

A string OAuth token from Outlook allowing access to a user's account

**class** `AutoReplyAudience`

**ALL** = 'All'

**CONTACTS\_ONLY** = 'ContactsOnly'

**INTERNAL\_ONLY** = 'None'

**class** `OutlookAccount.AutoReplyStatus`

**ALWAYS\_ENABLED** = 'AlwaysEnabled'

**DISABLED** = 'Disabled'

**SCHEDULED** = 'Scheduled'

`OutlookAccount.auto_reply_message`

The account's Internal auto reply message. Setting the value will change the auto reply message of the account, automatically setting the status to enabled (but not altering the schedule).

`OutlookAccount.deleted_messages()`

last ten deleted messages.

**Returns** list[Message]

`OutlookAccount.draft_messages()`

last ten draft messages.

**Returns** list[Message]

`OutlookAccount.get_folder_by_id(folder_id)`

`OutlookAccount.get_folders()`

Returns a list of all folders for this account

`OutlookAccount.get_message(message_id)`

Gets message matching provided id.

the Outlook email matching the provided `message_id`.

**Parameters** `message_id` – A string for the intended message, provided by Outlook

**Returns** Message

`OutlookAccount.get_messages(page=0)`

Get first 10 messages in account, across all folders.

**Keyword Arguments** `page` (*int*) – Integer representing the ‘page’ of results to fetch

**Returns** List[Message]

`OutlookAccount.inbox()`

first ten messages in account’s inbox.

**Returns** List[Message]

`OutlookAccount.new_email(body='', subject='', to=<class ‘list’>)`

Creates a NewMessage object.

**Keyword Arguments**

- **body** (*str*) – The body of the email
- **subject** (*str*) – The subject of the email
- **to** (*list* [Contact]) – A list of recipients to email

**Returns** Message

`OutlookAccount.send_email(body=None, subject=None, to=<class ‘list’>, cc=None, bcc=None, send_as=None, attachments=None)`

Sends an email in one method using variables to set the various pieces of the email.

**Parameters**

- **body** (*str*) – The body of the email
- **subject** (*str*) – The subject of the email
- **to** (*list*) – A list of *Contacts*
- **cc** (*list*) – A list of *Contacts* which will be added to the ‘Carbon Copy’ line
- **bcc** (*list*) – A list of *Contacts* while be blindly added to the email
- **send\_as** (*Contact*) – A *Contact* whose email the OutlookAccount has access to
- **attachments** (*list*) – A list of dictionaries with two parts [1] ‘name’ - a string which will become the file’s name [2] ‘bytes’ - the bytes of the file.

`OutlookAccount.sent_messages()`

last ten sent messages.

**Returns** list[Message]

`OutlookAccount.set_auto_reply(message, status='AlwaysEnabled', start=None, end=None, external_message=None, audience='All')`

Set an automatic reply for the account. :param message: The message to be sent in replies. If external\_message is provided this is the message sent to

internal recipients

### Parameters

- **status** – Whether the auto-reply should be always enabled, scheduled, or disabled. You can use [AutoReplyStatus](#) to provide the value. Defaults to ALWAYS\_ENABLED.
- **start** – If status is set to SCHEDULED, this is when the replies will start being sent.
- **end** – If status is set to SCHEDULED, this is when the replies will stop being sent.
- **external\_message** – If provided, this message will be sent to external recipients.
- **audience** – Whether replies should be sent to everyone, contacts only, or internal recipients only. You can use [AutoReplyAudience](#) to provide the value.

## Message

**class** `pyOutlook.core.message.Message` (*account, body, subject, to\_recipients, sender=None, cc=None, bcc=None, message\_id=None, \*\*kwargs*)

An object representing an email inside of the OutlookAccount.

**message\_id**

A string provided by Outlook identifying this specific email

**body**

The body content of the email, including HTML formatting

**body\_preview**

“The first 255 characters of the body”

**subject**

The subject of the email

**sender**

The [Contact](#) who sent this email. You can set this before sending an email to change which account the email comes from (so long as the [OutlookAccount](#) specified has access to the email).

**to**

A list of [Contacts](#). You can also provide a list of strings, however these will be turned into [Contacts](#) after sending the email.

**cc**

A list of [Contacts](#) in the CC field. You can also provide a list of strings, however these will be turned into [Contacts](#) after sending the email.

**bcc**

A list of [Contacts](#) in the BCC field. You can also provide a list of strings, however these will be turned into [Contacts](#) after sending the email.

**is\_draft**

Whether or not the email is a draft.

**importance**

The importance level of the email; with 0 indicating low, 1 indicating normal, and 2 indicating high. `Message.IMPORTANCE_LOW`, `Message.IMPORTANCE_NORMAL`, & `Message.IMPORTANCE_HIGH` can be used to reference the levels.

**categories**

A list of strings, where each string is the name of a category.

**time\_created**

A datetime representing the time the email was created

**time\_sent**

A datetime representing the time the email was sent

**IMPORTANCE\_HIGH = 2**

**IMPORTANCE\_LOW = 0**

**IMPORTANCE\_NORMAL = 1**

**add\_category** (*category\_name*)

**attach** (*file\_bytes*, *file\_name*)

Adds an attachment to the email. The filename is passed through Django's `get_valid_filename` which removes invalid characters. From the documentation for that function:

```
>>> get_valid_filename("john's portrait in 2004.jpg")
'johns_portrait_in_2004.jpg'
```

#### Parameters

- **file\_bytes** – The bytes of the file to send (if you send a string, ex for CSV, pyOutlook will attempt to convert that into bytes before base64ing the content).
- **file\_name** – The name of the file, as a string and leaving out the extension, that should be sent

**copy\_to** (*folder\_id*)

Copies the email to the folder specified by the *folder\_id*.

The folder id must match the id provided by Outlook.

**Parameters** **folder\_id** – A string containing the folder ID the message should be copied to

**copy\_to\_deleted** ()

Copies Message to account's Deleted Items folder

**copy\_to\_drafts** ()

Copies Message to account's Drafts folder

**copy\_to\_inbox** ()

Copies Message to account's Inbox

**delete** ()

Deletes the email

**forward** (*to\_recipients*, *forward\_comment=None*)

Forward Message to recipients with an optional comment.

#### Parameters

- **to\_recipients** – A list of *Contacts* to send the email to.
- **forward\_comment** – String comment to append to forwarded email.

### Examples

```
>>> john = Contact('john.doe@domain.com')
>>> betsy = Contact('betsy.donalds@domain.com')
>>> email = Message()
>>> email.forward([john, betsy])
>>> email.forward([john], 'Hey John')
```



**is\_read**

Set and retrieve the ‘Read’ status of an email

```
>>> message = Message()
>>> message.is_read
>>> False
>>> message.is_read = True
```

**move\_to(folder)**

Moves the email to the folder specified by the folder parameter.

**Parameters** **folder** – A string containing the folder ID the message should be moved to, or a Folder instance

**move\_to\_deleted()**

Moves the email to the account’s Deleted Items folder

**move\_to\_drafts()**

Moves the email to the account’s Drafts folder

**move\_to\_inbox()**

Moves the email to the account’s Inbox

**parent\_folder**

Returns the *Folder* this message is in

```
>>> account = OutlookAccount('')
>>> message = account.get_messages()[0]
>>> message.parent_folder
Inbox
>>> message.parent_folder.unread_count
19
```

Returns: *Folder*

**reply(reply\_comment)**

Reply to the Message.

**Notes**

HTML can be inserted in the string and will be interpreted properly by Outlook.

**Parameters** **reply\_comment** – String message to send with email.

**reply\_all(reply\_comment)**

Replies to everyone on the email, including those on the CC line.

With great power, comes great responsibility.

**Parameters** **reply\_comment** – The string comment to send to everyone on the email.

**send(content\_type='HTML')**

Takes the recipients, body, and attachments of the Message and sends.

**Parameters** **content\_type** – Can either be ‘HTML’ or ‘Text’, defaults to HTML.

## Folder

**class** `pyOutlook.core.folder.Folder` (*account*, *folder\_id*, *folder\_name*, *parent\_id*,  
*child\_folder\_count*, *unread\_count*, *total\_items*)

An object representing a Folder in the OutlookAccount provided.

**account**

The *OutlookAccount* this folder should be associated with

**id**

The static id generated by Outlook to identify this folder.

**folder\_name**

The name of this folder as displayed in the account

**parent\_id**

The id of the folder which houses this Folder object

**child\_folder\_count**

The number of child folders inside this Folder

**unread\_count**

The number of unread messages inside this Folder

**total\_items**

A sum of all items inside Folder

**copy\_into** (*destination\_folder*)

Copies the Folder into the provided destination folder.

**Raises** `AuthError` – Raised if Outlook returns a 401, generally caused by an invalid or expired access token.

**Parameters** `destination_folder` – The Folder that this Folder should be copied to.

**Returns** A new *Folder* representing the newly created folder.

**create\_child\_folder** (*folder\_name*)

Creates a child folder within the Folder it is called from and returns the new Folder object.

**Parameters** `folder_name` – The name of the folder to create

Returns: *Folder*

**delete** ()

Deletes this Folder.

**Raises** `AuthError` – Raised if Outlook returns a 401, generally caused by an invalid or expired access token.

**get\_subfolders** ()

Retrieve all child Folders inside of this Folder.

**Raises** `AuthError` – Raised if Outlook returns a 401, generally caused by an invalid or expired access token.

**Returns** `List[Folder]`

**messages** ()

Retrieves the messages in this Folder, returning a list of *Messages*.

**move\_into** (*destination\_folder*)

Move the Folder into a different folder.

This makes the Folder provided a child folder of the *destination\_folder*.

**Raises** `AuthError` – Raised if Outlook returns a 401, generally caused by an invalid or expired access token.

**Parameters** `destination_folder` – A *Folder* that should become the parent

**Returns** A new *Folder* that is now inside of the `destination_folder`.

**rename** (*new\_folder\_name*)

Renames the Folder to the provided name.

**Parameters** `new_folder_name` – A string of the replacement name.

**Raises** `AuthError` – Raised if Outlook returns a 401, generally caused by an invalid or expired access token.

**Returns** A new Folder representing the folder with the new name on Outlook.

## Contact

**class** `pyOutlook.core.contact.Contact` (*email, name=None*)

Represents someone sending or receiving an email. Cuts down on the amount of dictionaries floating around that each hold the API's syntax and allows for functionality to be added in the future.



## CHAPTER 5

---

### Indices and tables

---

- `genindex`
- `search`



## A

access\_token (OutlookAccount attribute), 9  
account (Folder attribute), 14  
add\_category() (pyOutlook.core.message.Message method), 12  
ALL (pyOutlook.core.main.OutlookAccount.AutoReplyAudience attribute), 9  
ALWAYS\_ENABLED (pyOutlook.core.main.OutlookAccount.AutoReplyStatusdraft\_messages() attribute), 9  
attach() (pyOutlook.core.message.Message method), 12  
auto\_reply\_message (pyOutlook.core.main.OutlookAccount attribute), 9

## B

bcc (Message attribute), 11  
body (Message attribute), 11  
body\_preview (Message attribute), 11

## C

categories (Message attribute), 11  
cc (Message attribute), 11  
child\_folder\_count (Folder attribute), 14  
Contact (class in pyOutlook.core.contact), 15  
CONTACTS\_ONLY (pyOutlook.core.main.OutlookAccount.AutoReplyAudience attribute), 9  
copy\_into() (pyOutlook.core.folder.Folder method), 14  
copy\_to() (pyOutlook.core.message.Message method), 12  
copy\_to\_deleted() (pyOutlook.core.message.Message method), 12  
copy\_to\_drafts() (pyOutlook.core.message.Message method), 12  
copy\_to\_inbox() (pyOutlook.core.message.Message method), 12  
create\_child\_folder() (pyOutlook.core.folder.Folder method), 14

## D

delete() (pyOutlook.core.folder.Folder method), 14  
delete() (pyOutlook.core.message.Message method), 12  
deleted\_messages() (pyOutlook.core.main.OutlookAccount method), 9  
DISABLED (pyOutlook.core.main.OutlookAccount.AutoReplyStatus attribute), 9  
draft\_messages() (pyOutlook.core.main.OutlookAccount method), 9

## F

Folder (class in pyOutlook.core.folder), 14  
folder\_name (Folder attribute), 14  
forward() (pyOutlook.core.message.Message method), 12

## G

get\_folder\_by\_id() (pyOutlook.core.main.OutlookAccount method), 9  
get\_folders() (pyOutlook.core.main.OutlookAccount method), 9  
get\_message() (pyOutlook.core.main.OutlookAccount method), 9  
get\_messages() (pyOutlook.core.main.OutlookAccount method), 10  
get\_subfolders() (pyOutlook.core.folder.Folder method), 14

## I

id (Folder attribute), 14  
importance (Message attribute), 11  
IMPORTANCE\_HIGH (pyOutlook.core.message.Message attribute), 12  
IMPORTANCE\_LOW (pyOutlook.core.message.Message attribute), 12  
IMPORTANCE\_NORMAL (pyOutlook.core.message.Message attribute), 12

inbox() (pyOutlook.core.main.OutlookAccount method), 10

INTERNAL\_ONLY (pyOutlook.core.main.OutlookAccount.AutoReplyAudience attribute), 9

is\_draft (Message attribute), 11

is\_read (pyOutlook.core.message.Message attribute), 12

## M

Message (class in pyOutlook.core.message), 11

message\_id (Message attribute), 11

messages() (pyOutlook.core.folder.Folder method), 14

move\_into() (pyOutlook.core.folder.Folder method), 14

move\_to() (pyOutlook.core.message.Message method), 13

move\_to\_deleted() (pyOutlook.core.message.Message method), 13

move\_to\_drafts() (pyOutlook.core.message.Message method), 13

move\_to\_inbox() (pyOutlook.core.message.Message method), 13

## N

new\_email() (pyOutlook.core.main.OutlookAccount method), 10

## O

OutlookAccount (class in pyOutlook.core.main), 9

OutlookAccount.AutoReplyAudience (class in pyOutlook.core.main), 9

OutlookAccount.AutoReplyStatus (class in pyOutlook.core.main), 9

## P

parent\_folder (pyOutlook.core.message.Message attribute), 13

parent\_id (Folder attribute), 14

## R

rename() (pyOutlook.core.folder.Folder method), 15

reply() (pyOutlook.core.message.Message method), 13

reply\_all() (pyOutlook.core.message.Message method), 13

## S

SCHEDULED (pyOutlook.core.main.OutlookAccount.AutoReplyStatus attribute), 9

send() (pyOutlook.core.message.Message method), 13

send\_email() (pyOutlook.core.main.OutlookAccount method), 10

sender (Message attribute), 11

sent\_messages() (pyOutlook.core.main.OutlookAccount method), 10

set\_auto\_reply() (pyOutlook.core.main.OutlookAccount method), 10

subject (Message attribute), 11

## T

time\_created (Message attribute), 11

time\_sent (Message attribute), 11

to (Message attribute), 11

total\_items (Folder attribute), 14

## U

unread\_count (Folder attribute), 14