
yagmail Documentation

Release 0.10.189

kootenpv

Sep 28, 2022

Contents

1	API Reference	3
1.1	Authentication	3
1.2	SMTP Client	3
1.3	E-Mail Contents	3
1.4	Exceptions	3
1.5	Utilities	3
2	Setup	5
2.1	Installing from PyPI	5
2.2	Installing from GitHub	5
2.3	Configuring Credentials	5
2.4	Using OAuth2	6
3	Usage	7
3.1	Start a Connection	7
3.2	Closing and reusing the Connection	7
3.3	Sending E-Mails	8
3.4	E-Mail recipients	8
3.5	Magical <code>contents</code>	8
3.6	Using yagmail from the command line	9
4	Indices and tables	11

yagmail is a GMAIL/SMTP client that aims to make it as simple as possible to send emails.

This page displays a full reference of *yagmail*'s API.

1.1 Authentication

Another way of authenticating is by passing an `oauth2_file` to `yagmail.SMTP`, which is among the safest methods of authentication. Please see the [OAuth2 section](#) of the [README](#) for further details.

It is also possible to simply pass the password to `yagmail.SMTP`. If no password is given, *yagmail* will prompt the user for a password and then store the result in the keyring.

1.2 SMTP Client

1.3 E-Mail Contents

1.4 Exceptions

1.5 Utilities

This page shows you how to install `yagmail` and how to set it up to use your system keyring service.

2.1 Installing from PyPI

The usual way of installing `yagmail` is through PyPI. It is recommended to install it together with the `keyring` library, by running the following (for Python 2.x and 3.x respectively):

```
pip install yagmail[all]
pip3 install yagmail[all]
```

If installing `yagmail` with `keyring` causes issues, omit the `[all]` to install it without.

2.2 Installing from GitHub

If you're not scared of things occasionally breaking, you can also install directly from the GitHub [repository](#). You can do this by running the following (for Python 2.x and 3.x respectively):

```
pip install -e git+https://github.com/kootenpv/yagmail#egg=yagmail[all]
pip3 install -e git+https://github.com/kootenpv/yagmail#egg=yagmail[all]
```

Just like with the PyPI installation method, if installing with `keyring` causes issues, simply omit the `[all]` to install `yagmail` without it.

2.3 Configuring Credentials

While it's possible to put the username and password for your E-Mail address into your script, `yagmail` enables you to omit both. Quoting from `keyrings` [README](#):

What is Python keyring lib?

The Python keyring lib provides a easy way to access the system keyring service from python. It can be used in any application that needs safe password storage.

If this sparked your interest, set up a Python interpreter and run the following to register your GMail credentials with yagmail:

```
import yagmail
yagmail.register('mygmailusername', 'mygmailpassword')
```

(this is just a wrapper for `keyring.set_password('yagmail', 'mygmailusername', 'mygmailpassword')`) Now, instantiating `yagmail.SMTP` is as easy as doing:

```
yag = yagmail.SMTP('mygmailusername')
```

If you want to also omit your username, you can create a `.yagmail` file in your home folder, containing just your username. Then, you can instantiate the SMTP client without passing any arguments.

2.4 Using OAuth2

Another fairly safe method for authenticating using OAuth2, since you can revoke the rights of tokens. In order to use OAuth2, pass the location of the credentials file to `yagmail.SMTP`:

```
yag = yagmail.SMTP('user@gmail.com', oauth2_file='~/oauth2_creds.json')
yag.send(subject="Great!")
```

If the file could not be found, then it will prompt for a `google_client_id` and `google_client_secret`. You can obtain these on [this OAuth2 Guide](#), upon which the OAuth2 code of yagmail is heavily based. After you have provided these, a link will be shown in the terminal that you should follow to obtain a `google_refresh_token`. Paste this again, and you're set up!

If somebody obtains the file, they can send E-Mails, but nothing else. As soon as you notice, you can simply disable the token.

This document aims to show how to use `yagmail` in your programs. Most of what is shown here is also available to see in the [README](#), some content may be duplicated for completeness.

3.1 Start a Connection

As mentioned in *Configuring Credentials*, there are three ways to initialize a connection by instantiating `yagmail.SMTP`:

1. **With Username and Password:** e.g. `yagmail.SMTP('mygmailusername', 'mygmailpassword')`
This method is not recommended, since you would be storing the full credentials to your account in your script in plain text. A better alternative is using `keyring`, as described in the following section:
2. **With Username and `keyring`:** After registering a `keyring` entry for `yagmail`, you can instantiate the client by simply passing your username, e.g. `yagmail.SMTP('mygmailusername')`.
3. **With `keyring` and `.yagmail`:** As explained in the *Setup* documentation, you can also omit the username if you have a `.yagmail` file in your home folder, containing just your GMail username. This way, you can initialize `yagmail.SMTP` without any arguments.
4. **With `OAuth2`:** This is probably the safest method of authentication, as you can revoke the rights of tokens. To initialize with `OAuth2` credentials (after obtaining these as shown in *Setup*), simply pass an `oauth2_file` to `yagmail.SMTP`, for example `yagmail.SMTP('user@gmail.com', oauth2_file='~/oauth2_creds.json')`.

3.2 Closing and reusing the Connection

By default, `yagmail.SMTP` will clean up after itself in **CPython**. This is an implementation detail of CPython and as such may not work in other implementations such as PyPy (reported in [issue #39](#)). In those cases, you can use `yagmail.SMTP` with `with` instead.

Alternatively, you can close and re-use the connection with `yagmail.SMTP.close()` and `yagmail.SMTP.login()` (or `yagmail.SMTP.oauth2_file()` if you are using OAuth2).

3.3 Sending E-Mails

`yagmail.SMTP.send()` is a fairly versatile method that allows you to adjust more or less anything about your Mail. First of all, all parameters for `yagmail.SMTP.send()` are optional. If you omit the recipient (specified with `to`), you will send an E-Mail to yourself.

Since the use of the (keyword) arguments of `yagmail.SMTP.send()` are fairly obvious, they will simply be listed here:

- `to`
- `subject`
- `contents`
- `attachments`
- `cc`
- `bcc`
- `preview_only`
- `headers`

Some of these - namely `to` and `contents` - have some magic associated with them which will be outlined in the following sections.

3.4 E-Mail recipients

You can send an E-Mail to a single user by simply passing a string with either a GMail username (@gmail.com will be appended automatically), or with a full E-Mail address:

```
yag.send(to='mike@gmail.com', contents="Hello, Mike!")
```

Alternatively, you can send E-Mails to a group of people by either passing a list or a tuple of E-Mail addresses as `to`:

```
yag.send(to=['to@someone.com', 'for@someone.com'], contents="Hello there!")
```

These E-Mail addresses were passed without any aliases. If you wish to use aliases for the E-Mail addresses, provide a dictionary mapped in the form `{address: alias}`, for example:

```
recipients = {
    'aliased@mike.com': 'Mike',
    'aliased@fred.com': 'Fred'
}
yag.send(to=recipients, contents="Hello, Mike and Fred!")
```

3.5 Magical contents

The `contents` argument of `yagmail.SMTP.send()` will be smartly guessed. You can pass it a string with your contents or a list of elements which are either:

- If it is a **dictionary**, then it will be assumed that the key is the content and the value is an alias (currently, this only applies to images). For example:

```
contents = [  
    "Hello Mike! Here is a picture I took last week:",  
    {'path/to/my/image.png': 'PictureForMike'}  
]
```

- If it is a **string**, then it will first check whether the content of the string can be **read as a file** locally, for example 'path/to/my/image.png'. These files require an extension for their content type to be inferred.
- If it could not be read locally, then it checks whether the string is valid HTML, such as `<h1>This is a big title!</h1>`.
- If it was not valid HTML either, then it must be text, such as "Hello, Mike!".

If you want to **ensure that a string is treated as text** and should not be checked for any other content as described above, you can use `yagmail.raw`, a subclass of `str`.

If you intend to **inline an image instead of attaching it**, you can use `yagmail.inline`.

3.6 Using yagmail from the command line

`yagmail` includes a command-line application, simply called with `yagmail` after you installed it. To view a full reference on how to use this, run `yagmail --help`.

CHAPTER 4

Indices and tables

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