pypersonalassistant Documentation

Release 0.1.2

Tom Milligan

Sep 27, 2017

Contents

1 About **1**

2 Installation **3**

3 Examples **5**
- 3.1 Communication: simple, secure messaging . . . 5
- 3.2 Logging: for easy python debugging . . . 6

4 Reference **7**
- 4.1 `helpers` Module . . . 7
- 4.2 `secure` Module . . . 8

5 Indices and tables **11**

Python Module Index **13**

CHAPTER 1

About

Aboutaboutabout

CHAPTER 2

Installation

```
pip install pypersonalassistant
```

PyCrypto needs to be compiled during installation; if this is a problem, install a prebuilt binary.

CHAPTER 3

Examples

Communication: simple, secure messaging

Send some emails

The following script allows a user to enter a master password before starting a long task, authorising later use of their email (gmail) account.

```
import pypersonalassistant.secure

ppa_secure = pypersonalassistant.secure.secure()
results = some_long_task(foo, bar)
recipients = ['my.boss@company.com',
              'coworker.one@company.com',
              'coworker.two@company.com'
              ]
content = 'Subject: Results of my task\n{0}'.format(results)
for r in recipients:
    ppa_secure.email(r, content)
```

Send an SMS

Or send SMS messages using (twilio).

```
import pypersonalassistant.secure

ppa_secure = pypersonalassistant.secure.secure()
some_long_task(foo, bar)
ppa_secure.SMS_me('Task completed')
```

Logging: for easy python debugging

Start logging

The following script allows a user to easily start logging

- warnings to stdout
- debug to a file

Logging is triggered using the using the standard logging module.

```
import logging
import pypersonalassistant.helpers

ppa_logger = pypersonalassistant.helpers.log()
logging.debug('START LOG')
```

Stop/pause logging

Logging can be easily disabled at any time

```
ppa_logger.disable()
...
ppa_logger.enable()
```

Customise logging

Details of the underlying logger object can be read or modified to change behaviour, via the `logger` attribute

```
ppa_logger.logger.setLevel(logging.INFO)
```

CHAPTER 4

Reference

helpers Module

The *helpers* module provides useful classes and functions that are not part of pypersonalassistant's core functionality.

log Class

class `pypersonalassistant.helpers.log`(*logfile=None*, *file_level=10*, *stream_level=30*)

Easily setup logs using the standard logging module.

Parameters

- **logfile** (*string*) – Path to the log file. Defaults to <your_script>_py.log
- **file_level** – The logging level to log to file
- **stream_level** – The logging level to log to stdout

Returns A *pypersonalassistant.helpers.log* object

Methods

`log.enable()`

Enable/re-enable logging

`log.disable()`

Disable logging

Attributes

`log.logger`

The logger object listening for logging calls; can be used to further customise logging

Type logging.logger

`secure` Module

The `secure` module provides a class with the same name which is used to represent a user's consent to pypersonalassistant using their personal credentials.

secure Class

class `pypersonalassistant.secure.`**`secure`**(*credentials_path='credentials.json'*)

This class represents a user's consent to pypersonalassistant using their personal credentials in the future.

Note: On creating an instance of this object, user input is required.

Asks the user for the password used to previously encrypt their private credentials. If it is the first time the module has been used, the password will be used to encrypt credentials entered in future.

Parameters **`credentials_path`** (*`string`*) – The file used to retrieve/store encrypted personal credentials.

Returns A *`pypersonalassistant.secure.secure`* object

Methods

An instance of the *`secure`* class has the following methods.

Email

`secure.`**`gmail_email`**(*from_*, *to*, *msg*)

Send an email from your gmail account.

msg can either be:

- A string, in which case:
  - At the first newline (n) the string will be split into subject and body
  - If no newline is present, the entire string will be body.
- An email.message.Message object

Login will be performed using stored credentials.

- *stored credential name: GMAIL_EMAIL*
- *stored credential name: GMAIL_EMAIL_PASSWORD*

Parameters

- **`from`** (*`string`*) – The phone number in your twilio account to send the SMS message from. Full international format.
- **`to`** (*`string`*) – The email address to send the email to.
- **`body`** – The content of the email. See above.

`secure.email`(*to*, *msg*)

Quickly send an email from a default address. Calls `gmail_email()`.

- *stored credential name: GMAIL_EMAIL*

Parameters

- **to** (*string*) – The email address to send the email to.
- **msg** – The content of the email. See `gmail_email()`.

`secure.email_me`(*msg*)

Quickly send an email to yourself. Calls `email()`.

- *stored credential name: PERSONAL_EMAIL*

Parameters **msg** – The content of the email. See `gmail_email()`.

SMS

`secure.twilio_SMS`(*from_*, *to*, *body*)

Send an SMS message from your twilio account.

Login will be performed using stored credentials.

- *stored credential name: TWILIO_ACCOUNT_SID*
- *stored credential name: TWILIO_AUTH_TOKEN*

Parameters

- **from** (*string*) – The phone number in your twilio account to send the SMS message from. Full international format.
- **to** (*string*) – The phone number to send the SMS message to. Full international format.
- **body** (*string*) – The content of the SMS message.

`secure.SMS`(*to*, *body*)

Quickly send an SMS from a default number. Calls `twilio_SMS()`.

- *stored credential name: TWILIO_PHONE_NUMBER*

Parameters

- **to** (*string*) – The phone number to send the SMS message to. Full international format.
- **body** (*string*) – The content of the SMS message.

`secure.SMS_me`(*body*)

Quickly send an SMS to yourself. Calls `SMS()`.

- *stored credential name: PERSONAL_PHONE_NUMBER*

Parameters **body** (*string*) – The content of the SMS message.

Raw credentials

`secure.`**`credential`**(*key*)

Returns the decrypted (plain-text) value of the credential specified.

Parameters **`key`** (*string*) – The credential name.

Returns The credential value as a string.

`secure.`**`edit_credentials`**(*credential=None*, *already_set=True*)

Edit the credentials used by the *secure* class.

Note: User input is required.

By default, unset credentials will be displayed.

Parameters

- **`credential`** (*string*) – A specific credential to edit.
- **`already_set`** (*bool*) – If *False* will show all credentials, whether set or unset.

Attributes

`secure.`**`credentials_path`**

The path to the file containing encrypted personal credentials

Type string

CHAPTER 5

Indices and tables

- genindex
- modindex
- search

Python Module Index

p

pypersonalassistant.helpers, 7
pypersonalassistant.secure, 8

Index

C

credential() (pypersonalassistant.secure.secure method), 10
credentials_path (pypersonalassistant.secure.secure attribute), 10

D

disable() (pypersonalassistant.helpers.log method), 7

E

edit_credentials() (pypersonalassistant.secure.secure method), 10
email() (pypersonalassistant.secure.secure method), 8
email_me() (pypersonalassistant.secure.secure method), 9
enable() (pypersonalassistant.helpers.log method), 7

G

gmail_email() (pypersonalassistant.secure.secure method), 8

L

log (class in pypersonalassistant.helpers), 7
logger (pypersonalassistant.helpers.log attribute), 7

P

pypersonalassistant.helpers (module), 7
pypersonalassistant.secure (module), 8

S

secure (class in pypersonalassistant.secure), 8
SMS() (pypersonalassistant.secure.secure method), 9
SMS_me() (pypersonalassistant.secure.secure method), 9

T

twilio_SMS() (pypersonalassistant.secure.secure method), 9