pygett Documentation

Release 1.0

Mark Allen

December 13, 2011

CONTENTS

CHAPTER ONE

ABOUT

This library provides a binding to the REST API for the file sharing service Ge.tt. Please see the Ge.tt Developer's Documentation for information on how to get an API key.

CHAPTER

TWO

INSTALLATION

To install, use the standard `python setup.py install`

CHAPTER

THREE

QUICK USAGE

The API initializaton requires the following parameters to be present:

- **apikey**: The API key assigned by Ge.tt for your application
- **email**: The email address linked to an API key
- **password**: The password linked to an API key

Example initialization:

```
from pygett import Gett

client = Gett(
        apikey = "apitest",
        email = "apitest@ge.tt",
        password = "secret"
        )
```

Getting a dict of all shares:

```
shares = client.get_shares()
for file in shares['4ddfds'].files
    print file.filename
```

Getting a list of all shares:

```
shares = client.get_shares_list()
for share in shares:
    for file in share.files:
        print share.sharename + "\t" + file.filename + "\t" + file.size
```

Getting a specific share:

```
share = client.get_share("4ddfds")
```

Getting a specific file:

```
file = client.get_file("4ddfds", 0)
```

Uploading a file:

```
file = client.upload_file(
        filename = "test.rst",
        data = open("test.rst", "rb").read()
        )

print "File '%s' is now available at %s" % (file.filename, file.getturl)
```

Downloading file content:

```
file = client.get_file("4ddfds", 0)
buffer = file.contents()
```

Most methods return a `pygett.base.Gett` specific object such as `pygett.shares.GettShare`, `pygett.files.GettFile` or `pygett.user.GettUser`.

CHAPTER

FOUR

CONTENTS

4.1 pygett Package

4.1.1 `base` Module

class `pygett.base.`**`Gett`**(**args*, ***kwargs*)

Base client object

Requires the following keyword arguments:

- `apikey` - The API key assigned to an application by Gett
- `email` - The email address linked to the API key
- `password` - The password linked to the API key

Attribute

- `user` - a `pygett.user.GettUser` object

`create_share`(***kwargs*)

Create a new share. Takes a keyword argument.

Input:

- `title` optional share title (optional)

Output:

- A `pygett.shares.GettShare` object

Example:

```
new_share = client.create_share( title="Example Title" )
```

`get_file`(*sharename*, *fileid*)

Get a specific file. Does not require authentication.

Input:

- A sharename
- A fileid - must be an integer

Output:

- A `pygett.files.GettFile` object

Example:

```
file = client.get_file("4ddfds", 0)
```

get_share(*sharename*)
Get a specific share. Does not require authentication.

Input:

- A sharename

Output:

- A `pygett.shares.GettShare` object

Example:

```
share = client.get_share("4ddfds")
```

get_shares(***kwargs*)
Gets *all* shares.

Input:

- `skip` the number of shares to skip (optional)
- `limit` the maximum number of shares to return (optional)

Output:

- a dict where keys are sharenames and the values are corresponding `pygett.shares.GettShare` objects

Example:

```
shares = client.get_shares()
```

get_shares_list(***kwargs*)
Gets *all* shares.

Input:

- `skip` the number of shares to skip (optional)
- `limit` the maximum number of shares to return (optional)

Output:

- a list of `pygett.shares.GettShare` objects

Example:

```
shares_list = client.get_shares_list()
```

upload_file(***kwargs*)
Upload a file to the Gett service. Takes keyword arguments.

Input:

- `filename` the filename to use in the Gett service (required)
- `data` the file contents to store in the Gett service (required) - must be a string
- `sharename` the name of the share in which to store the data (optional); if not given, a new share will be created.
- `title` the share title to use if a new share is created (optional)

Output:

- A `pygett.files.GettFile` object

Example:

```
file = client.upload_file(filaname="foo", data=open("foo.txt").read())
```

4.1.2 `exceptions` Module

exception `pygett.exceptions.`**`GettError`**(*status_code*, *endpoint*, *params*)

Base error class

Attributes

- `http_status` The HTTP status code from the remote server
- `endpoint` The URI to which a request was attempted
- `error` A message describing the error

4.1.3 `files` Module

class `pygett.files.`**`GettFile`**(*user*, ***kwargs*)

Encapsulate a file in the Gett service.

Attributes

This object has the following attributes:

- `fileid` - A file id as assigned by the Gett service
- `sharename` - The sharename in which this file is contained
- `downloads` - The number of downloads of this file
- `getturl` - The URL at which this file can be viewed in a browser
- `filename` - The user specified filename
- `readystate` - The Gett state of this file

During file uploads, the following attributes will be set:

- `put_upload_url` - A URL suitable for use with the PUT HTTP verb (see `send_file()`)
- `post_upload_url` - A URL suitable for use with the POST HTTP verb

`contents`()

This method downloads the contents of the file represented by a *GettFile* object's metadata.

Input:

- None

Output:

- A byte stream

NOTE: You are responsible for handling any encoding/decoding which may be necessary.

Example:

```
file = client.get_file("4ddfds", 0)
print file.contents()
```

`destroy()`

This method removes the file's content and metadata from the Gett service. There is no way to recover the data once this method has successfully completed.

Input:

- None

Output:

- `True`

Example:

```
client.get_file("4ddfds", 0).destroy()
```

`refresh()`

Retrieve current file metadata from the Gett service.

Input:

- None

Output:

- None

Example:

```
file = client.get_file("4ddfds", 0)
print "File size: %s" % file.size  # File size: 96
file.send_data(put_url=file.upload_url, data=open("example.txt", "rb").read())
file.refresh()
print "File size: %s" % file.size  # File size: 109
```

`send_data(`***kwargs*`)`

This method transmits data to the Gett service.

Input:

- `put_url` A PUT url to use when transmitting the data (required)
- `data` A byte stream (required)

Output:

- `True`

Example:

```
if file.send_data(put_url=file.upload_url, data=open("example.txt", "rb").read()):
    print "Your file has been uploaded."
```

`thumbnail()`

This method returns a thumbnail representation of the file if the data is a supported graphics format.

Input:

- None

Output:

- A byte stream representing a thumbnail of a support graphics file

Example:

```
file = client.get_file("4ddfds", 0)
open("thumbnail.jpg", "wb").write(file.thumbnail())
```

upload_url()

This method generates URLs which allow overwriting a file's content with new content. The output is suitable for use in the `send_data()` method below.

Input:

- None

Output:

- A URL (string)

Example:

```
file = client.get_file("4ddfds", 0)
file.send_data(put_url=file.upload_url, data=open("example.txt", "rb").read())
```

4.1.4 `request` Module

class `pygett.request.`**BaseRequest**(**args*, ***kwargs*)

Base request class

get(*endpoint*, **args*, ***kwargs*)

get

Make a GET call to a remote endpoint

Input:

- An endpoint relative to the `base_url`

Output:

- A `pygett.request.GettResponse` object

post(*endpoint*, *d*, **args*, ***kwargs*)

post

Make a POST call to a remote endpoint

Input:

- An endpoint relative to the `base_url`
- POST data

NOTE: Passed POST data will be automatically serialized to a JSON string if it's not already a string

Output:

- A `pygett.request.GettResponse` object

put(*endpoint*, *d*, **args*, ***kwargs*)

put

Make a PUT call to a remove endpoint

Input:

- An absolute endpoint
- A data stream

Output:

- A `pygett.request.GettResponse` object

class `pygett.request.`**`GettRequest`**(**args*, ***kwargs*)

Encapsulate a request to the Gett service

Attributes

- `base_url` The base URL of the service (defaults to https://open.ge.tt/1)
- `endpoint` The full URL of the remote endpoint
- `type` The type of request (GET, POST, or PUT)
- `data` The data for a POST or PUT request

class `pygett.request.`**`GettResponse`**(*http_status*, *response*)

Encapsulate responses from the Gett service

Attributes

- `http_status` The status code from the remote endpoint
- `string_response` The response as a string, before deserialization
- `response` The response serialized into a dict

4.1.5 `shares` Module

class `pygett.shares.`**`GettShare`**(*user*, ***kwargs*)

Encapsulate a share in the Gett service.

Attributes

- `sharename` The sharename
- `title` The share title (if any)
- `created` Unix epoch seconds when the share was created
- `files` A list of all files contained in a share as `pygett.files.GettFile` objects

`destroy`()

This method removes this share and all of its associated files. There is no way to recover a share or its contents once this method has been called.

Input:

- None

Output:

- `True`

Example:

```
client.get_share("4ddfds").destroy()
```

`refresh`()

This method refreshes the object with current metadata from the Gett service.

Input:

- None

Output:

- None

Example:

```
share = client.get_share("4ddfds")
print share.files[0].filename      # prints 'foobar'
if share.files[0].destroy():
    share.refresh()
    print share.files[0].filename  # now prints 'barbaz'
```

`update`(***kwargs*)
Add, remove or modify a share's title.

Input:

- `title` The share title, if any (optional)

NOTE: Passing `None` or calling this method with an empty argument list will remove the share's title.

Output:

- None

Example:

```
share = client.get_share("4ddfds")
share.update(title="Example") # Set title to Example
share.update()                # Remove title
```

4.1.6 `user` Module

class `pygett.user.`**`GettUser`**(*apikey*, *email*, *password*)
Encapsulates Gett user functionality

Attributes

- `apikey` The API key assigned by Gett for an application
- `email` The email linked to the API key
- `password` The password linked to the API key

After a successful login the following attributes are populated:

- `refresh_token` Used to get a new valid access token without requiring the API key, email and password
- `access_token_expires` - Epoch seconds until the current access token is no longer valid. Typically 86400 seconds from login. (Suitable for use with `time.localtime()`)
- `access_token_grace` - How many seconds before an access token is scheduled to expire to attempt a renewal. (Defaults to 3600 seconds)
- `userid` - User ID string supplied by Gett
- `fullname` - The full name linked to an authenticated user account
- `storage_used` - The amount of storage consumed (in total) for this user account. (Unit: bytes)
- `storage_limit` - The maximum number of bytes available for storage. (Unit: bytes)

`access_token`()
access_token

Returns a valid access token. If the user is not currently logged in, attempts to do so. If the current time exceeds the grace period, attempts to retrieve a new access token.

Input:

- None

Output:

- A valid access token

Example:

```
print "Your access token is currently %s" % client.user.access_token()
```

login(***params*)

login

Use the current credentials to get a valid Gett access token.

Input:

- A dict of parameters to use for the login attempt (optional)

Output:

- `True`

Example:

```
if client.user.login():
    print "You have %s bytes of storage remaining." % ( client.user.storage_limit - client_u
```

refresh()

refresh

Refresh this user object with data from the Gett service

Input:

- None

Output:

- `True`

Example:

```
if client.user.refresh():
    print "User data refreshed!"
    print "You have %s bytes of storage remaining." % ( client.user.storage_limit - client_u
```

CHAPTER

FIVE

LICENSE

MIT License

Copyright (c) 2011 Mark Allen

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

CHAPTER

SIX

INDICES AND TABLES

- *genindex*
- *modindex*
- *search*

PYTHON MODULE INDEX