Desktoppr API Wrapper for Python3 Documentation

Release 1.0

Mgamerz

Jul 02, 2017

Contents

This is the Documentation for the Desktoppr API Wrapper for Python3. It allows you to easily work with the Desktoppr API with human readable commands and built-in error handling.

Contents:

CHAPTER 1

Prerequesites

Required Packages

Getting started with the Desktoppr API wrapper is easy. All you need is Python3 and the requests library.

Note: This wrapper was not written for Python 2.x. It does *not* work on Python 2.x.

If you aren't sure if you have requests, you can check by starting the python interpreter you're going to use and importing the requests module. If you get the following error below, you don't have the requests module installed:

```
$ python3
Python 3.2.3 (default, Sep 25 2013, 18:22:43)
[GCC 4.6.3] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import requests
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ImportError: No module named requests
>>>
```

You can install requests like this if you don't have it:

```
$ pip install requests
```

You may need to use your specific python's pip version if it defaults to a different version than the one you want:

```
$ pip3.3 install requests
```

API Authorization

To interact with the site as a user you will need to to make an account on the Desktoppr.co website. This will allow you to like and sync wallpapers, flag wallpapers, as well as other user-based tasks. If you aren't going to interact as a user, you won't need to authorize to the server.

CHAPTER 2

Basic Usage

The four object types

Before we dive into using the API, you should understand that the wrapper returns four object types:

- `Page` – Page of information, containing information about other pages and containing a list of either `Wallpaper` or `User` objects.
- `Wallpaper` – Object defining a wallpaper on the site, containing information about the image, thumbnail and preview image, and metadata about the image.
- `User` – Object defining a user on the site, containing metadata about them, including things like sign up date and likes count.
- `Image` – Object defining an image on the site. It is contained in a `Wallpaper` object. It contains the direct URLs to images (full, preview, and thumbnails)

Examples

A simple test

Let's start with a very simple test. You want to get the number of wallpapers the user *keithpitt* has in their collection. It's as simple as follows:

```
>>> import DesktopprApi
>>> api = DesktopprApi.DesktopprAPI()
>>> user = api.get_user_info('keithpitt')
>>> user.wallpapers_count
93
```

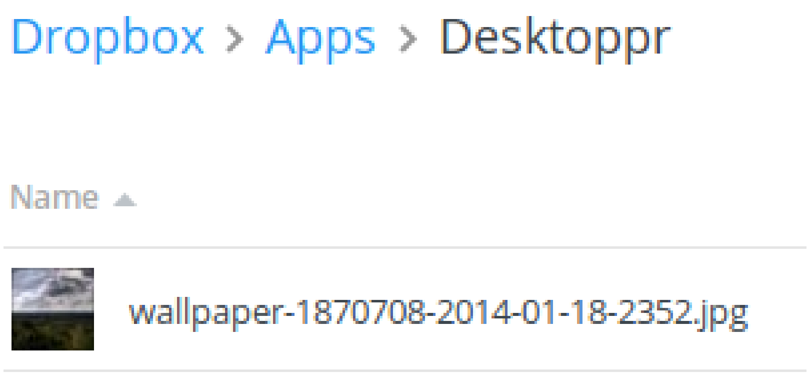
Each of the four object types have all attributes that the server responds with available by *objectname.attribute*.

Another simple example

We want to sync a wallpaper to our DropBox progmatically. In this instance, we want it to be random wallpaper that's safe for work. Assuming we already have linked a DropBox account and have our API key, we can do it as follows:

```
>>> import DesktopprApi
>>> api = DesktopprApi.DesktopprAPI()
>>> wallpaper = api.get_random_wallpaper()
>>> api.authorize_API('YOUR_API_KEY_HERE')
>>> api.sync_wallpaper(wallpaper.id)
```

And voilà, it works!


Dropbox › Apps › Desktoppr
Name
wallpaper-1870708-2014-01-18-2352.jpg

Getting straight URLs

The method *get_wallpaper_urls()* is a convenience method built into the wrapper that allows you to get direct URLs to full resolution images.

CHAPTER 3

Module Documentation

class `DesktopprApi.`**`DesktopprAPI`**

This class allows you to create an object that allows you to query the Desktoppr site using their public API.

`authorize_API`(*apikey*)

Authorizes using a users api key. This does not require the user's password or username.

Parameters **`apikey`** (*`str`*) – The api key for your user account. Successfully authorizing will store this key in the field self.apikey.

Returns

- **True** – if the authorization worked with the given apikey.
- **False** – if the authorization did not work with the given apikey.

`authorize_user_pass`(*username*, *password*)

Gets a users api key by authorizing to the site with a username/ password. Stores the users API key in the field self.apikey for further privileged access in this object's session.

Parameters

- **`username`** (*`str`*) – Username to login with.
- **`password`** (*`str`*) – User's password. You should take steps to secure the entry of this argument.

Returns

- **True** – if the authorization worked with the given username/password. The API key of the user is now cached.
- **False** – if the authorization did not work with the given username/password.

`check_if_liked`(*username*, *wallpaper_id*)

Checks if a user has liked a wallpaper.

Parameters

- **`username`** (*`str`*) – Username to check for liking a wallpaper

- **wallpaper_id** (*int*) – Wallpaper to check if the user likes it

Returns

- **None** – if an error occurs (user doesn't exist, etc).
- **True** – if the user has liked the wallpaper.
- **False** – if the user hasn't liked the wallpaper.

check_if_synced(*username*, *wallpaper_id*)

Checks if a user has a wallpaper currently synced to their personal DropBox.

Parameters

- **username** (*str*) – Username to check for wallpaper sync status on
- **wallpaper_id** (*int*) – Wallpaper to check if it is synced

Returns

- **True** – if the wallpaper exists in the user's DropBox since their last relink to DropBox. If a DropBox account is unlinked and relinked, all previous synced wallpapers are ignored, even if they still reside in their DropBox folder.
- **False** – If the wallpaper is not synced, or if an error occurs.

flag_wallpaper(*wallpaper_id*, *flag*)

Flags a wallpaper for filtering on the site.

> **Warning:** This is a privileged method. You must authorize with *authorize_user_pass()* or *authorize_API()* before you can use it.

Parameters

- **wallpaper_id** (*int*) – id of the wallpaper.
- **flag** (*str*) – Flag to place on a wallpaper. Valid options are **flag_safe**, **flag_not_safe**, and **flag_deletion**

Returns

- **None** – if the you haven't authorized against the server yet or if the flag is invalid.
- **True** – if the Wallpaper was successfully flagged.
- **False** – if the Wallpaper was not successfully flagged.

follow_user(*username*)

Attempts to follow a user.

> **Warning:** This is a privileged method. You must authorize with *authorize_user_pass()* or *authorize_API()* before you can use it.

Parameters **username** (*str*) – Username to follow

Returns

- **None** – if the you haven't authorized against the server yet.
- **True** – if the follow attempt succeeded.

- **False** – if the follow attempt failed.

`get_followed_users` (*username*, *page=1*)
Gets a page containg a list of User objects who the specified user follows.

Parameters **`username`** (`str`) – Username to query.

Returns

- **None** – if the user follows noone, the user cannot be found, or an error occurs.
- `Page` object – if the command succeeds. The page object's username attribute is populated with the list of users.

`get_random_wallpaper` (*safefilter='safe'*)
Retrieves a random wallpaper.

The safefilter can return different levels of images:

```
safe = Safe for work
include_pending = Images not yet marked as safe or not safe for work (NSFW)
all = All images, including NSFW images
```

Parameters **`safefilter`** (`str`) – *Optional*, Safety filter for returned images. Valid strings are **safe**, **include_pending**, and **all**. Defaults to **safe**.

Returns

- **None** – if a bad safefilter is passed (if any) or there was an error getting a wallpaper.
- `Wallpaper` object – if successful.

`get_user_collection` (*username*, *page=1*)
Gets a page of wallpapers defining ones in a users collection.

Parameters

- **`username`** (`str`) – User to query for information
- **`page`** (`int`) – *Optional*, the page number to return. The number of results per page is limited by the server. The default is **page 1**.

Returns

- **None** – if an error occurs (no wallpapers, invalid user...)
- `Page` object – if the command succeeds. The page object's wallpapers attribute is populated with the list of wallpapers the user has in their collection.

`get_user_followers` (*username*, *page=1*)
Gets a `Page` contains a list of of `User` objects representing users who follow this user. The pages can be iterated over to find all followers.

Parameters

- **`username`** (`str`) – Username to query.
- **`page`** (`int`) – Page of results to return

Returns

- **None** – if the user has no followers, cannot be found, or an error occurs.
- `Page` object – if the command succeeds. The page object's users attribute is populated with the list of users.

get_user_info(*username*)

Get information about a user.

Parameters **username** (*str*) – User to query for information

Returns

- **None** if the request did not return user information.
- *User* object – if the command succeeds. The user object's attributes can be parsed for desired information.

get_user_randomwallpaper(*username*)

Fetches a random wallpaper a user has in their collection.

Parameters **username** (*str*) – Username to get random wallpaper from

Returns

- **None** – if a failure occurred trying to get a wallpaper
- *Wallpaper* object – If successful.

get_userlikes(*username*, *page=1*)

Gets a list of wallpapers that a user likes.

Parameters

- **username** (*str*) – Username to get list of liked wallpapers for
- **page** (*int*) – *Optional*, return different list of pages. Defaults to **page 1**.

Returns

- **None** – if an error occurs (not a user, etc).
- *Page* object – if successful. Wallpapers the user likes can be accessed via the .wallpapers attribute.

get_wallpaper_urls(*page=1*, *safefilter='safe'*)

This is a subset of *get_wallpapers()*, which returns a page of wallpaper URLs. The API does not document sorting options. It uses the same interface as get_wallpapers.

Parameters

- **page** (*int*) – *Optional*, page number to return. The server limits how many results are returned by query, so pages allow you to sift through results.
- **safefilter** (*str*) – *Optional*, safety filter for returned images. Valid strings are **safe**, **include_pending**, and **all**. Defaults to **safe**.

Returns

- *Page* object – if the command succeeds. The page object's wallpapers attribute is populated with the list of wallpapers.
- **None** – If an error occurs trying to get wallpapers.

get_wallpapers(*page=1*, *safefilter='safe'*)

Retrieves a list of wallpapers. The page parameter can query different pages of results.

The safefilter can return different levels of images:

```
safe = Safe for work
include_pending = Images not yet marked as safe or not safe for work (NSFW)
all = All images, including NSFW images
```

Parameters

- **page** (*int*) – *Optional*, page of results to retrieve. Defaults to **page 1**.
- **safefilter** (*str*) – *Optional*, Safety filter for returned images. Valid strings are **safe**, **include_pending**, and **all**. Defaults to **safe**.

Returns

- **None** if a bad safefilter is passed (if any) or there was an error getting wallpapers.
- *Page* object – if the command succeeds. The page object's wallpapers attribute is populated with the list of wallpapers the returned by the server.

like_wallpaper (*wallpaper_id*)

> **Warning:** This is a privileged method. You must authorize with *authorize_user_pass()* or *authorize_API()* before you can use it.

Likes a wallpaper.

Parameters **wallpaper_id** (*int*) – Wallpaper to like

Returns

- **None** – if the you haven't authorized against the server yet.
- **True** – if the like succeeded.
- **False** – if the like attempt failed.

sync_wallpaper (*wallpaper_id*)

Informs the server that it should start a sync of a wallpaper to a user's DropBox. This checks against the server for wallpapers.

> **Warning:** This is a privileged method. You must authorize with *authorize_user_pass()* or *authorize_API()* before you can use it.

Parameters **wallpaper_id** (*int*) – The wallpaper id to sync to the authorized user's DropBox

Returns

- **None** – if the you haven't authorized against the server yet.
- **True** – if a wallpaper was set to sync (or was already synced).
- **False** – if the HTTP response is not 200 or 422 (already synced)

unfollow_user (*username*)

Attempts to unfollow a user.

Parameters **username** (*str*) – User to unfollow

Returns

- **None** – if the you haven't authorized against the server yet.
- **True** – if the unfollow attempt succeeded.

- **False** – if the unfollow attempt failed.

unlike_wallpaper(*wallpaper_id*)
Unlikes a wallpaper.

> **Warning:** This is a privileged method. You must authorize with `authorize_user_pass()` or `authorize_API()` before you can use it.

Parameters **wallpaper_id** (*int*) – Wallpaper to unlike

Returns

- **None** – if the you haven't authorized against the server yet.
- **True** – if the unlike succeeded.
- ** False** – if the unlike attempt failed.

unsync_wallpaper(*wallpaper_id*)
Informs the server that it should remove a wallpaper from a user's DropBox. This checks against the user's DropBox for wallpapers, so if it doesn't exist there, it will still return **True**.

> **Warning:** This is a privileged method. You must authorize with `authorize_user_pass()` or `authorize_API()` before you can use it.

Parameters **wallpaper_id** (*int*) – The wallpaper id to unsync from the authorized user's DropBox

Returns

- **None** – if the you haven't authorized against the server yet.
- **True** – if a wallpaper was set to unsync (or did not exist).
- **False** – if the HTTP response is not 200 or 404 (Not in user's DropBox)

class DesktopprApi.**Image**(*info=None*)
Represents an image object (a part of a wallpaper object). All values are initialized to none, but are set if an image dict is passed.

An image object can contain another image object. There are only two different setups of an image object:

- Contains thumb, preview, and url = **Full Resolution Image**
- Contains width, height, and url = **Thumbnail or Preview Image**

class DesktopprApi.**Page**(*infotype*, *info*)
A page object represents a 'page' of information returned by the API when it involves paginated information. It contains either a list of `Wallpaper` objects or a list of `User` objects.

class DesktopprApi.**User**(*info=None*)
Defines a user on the site.

class DesktopprApi.**Wallpaper**(*info=None*)
Defines a Wallpaper on the Desktoppr server. Contains many attributes about the image.

CHAPTER 4

Indices and tables

- genindex
- modindex
- search

Python Module Index

Index