
Pybooru Documentation

Daniel Luque

Oct 17, 2020

Contents:

1	Dependencies	3
1.1	Quick Start Guide	3
1.2	Danbooru Reference	5
1.3	Danbooru API Reference	6
1.4	Moebooru Reference	20
1.5	Moebooru API Reference	22
1.6	Pybooru Reference	29
2	Changelog	33
	Python Module Index	35
	Index	37

Pybooru is a Python package to access to the API of Danbooru/Moebooru based sites.

- Version: **4.2.2**
- Licensed under: [MIT License](#)
- Python: ≥ 2.7 or Python: ≥ 3.3

Note: Pybooru 4.2.2 Is the last version that support Python 2.7

CHAPTER 1

Dependencies

- `requests`

1.1 Quick Start Guide

1.1.1 Features

- Support Danbooru API (version: 2.105.0 - 77e06b6).
- Support Moebooru API (version: 1.13.0+update.3).
- Default site list.
- JSON responses.
- Custom user-agent.

1.1.2 Installation

You can download and install Pybooru from [Pypi](#)

```
pip install --user Pybooru
```

1.1.3 Install from source

```
git clone git://github.com/luquedaniel/pybooru.git
cd pybooru
pip install --user -r requirements.txt
sudo python setup.py build
python setup.py install
```

1.1.4 Example of use with Danbooru

```
from pybooru import Danbooru

client = Danbooru('danbooru')
artists = client.artist_list('ma')

for artist in artists:
    print("Name: {}".format(artist['name']))
```

login (only for functions that **requires login**):

You can pass two parameters to authenticate: “username” and “api_key”. You can see your API key in your profile.

Example:

```
from pybooru import Danbooru

client = Danbooru('danbooru', username='your-username', api_key='your-apikey')
client.comment_create(post_id=id, body='Comment content')
```

1.1.5 Example of use with Moebooru

```
from pybooru import Moebooru

client = Moebooru('konachan')
artists = client.artist_list(name='ma')

for artist in artists:
    print("Name: {}".format(artist['name']))
```

Some functions may require you to authenticate:

- **username**: your site username.
- **password**: your password in plain text.
- **hash_string** (requires only for sites that isn’t in default site list): a string to be hashed with your password.

Example using default sites:

```
from pybooru import Moebooru

client = Moebooru('konachan', username='your-username', password='your-password')
client.comment_create(post_id=id, comment_body='Comment content')
```

Example using not default sites:

```
from pybooru import Moebooru

client = Moebooru('konachan.com', username='your-username', password='your-password',
                  hash_string='So-I-Heard-You-Like-Mupkids-?--{}--')
client.comment_create(post_id=id, comment_body='Comment content')
```

See more examples of [Danbooru](#) and [Moebooru](#).

1.1.6 Default sites list

Pybooru has a list of default sites that allow you to use Pybooru without “site_url” argument:

- konachan ([Konachan](#))
- yandere ([Yande.re](#))
- danbooru ([Danbooru](#))
- safebooru ([Safebooru](#))
- lolibooru ([Lolibooru](#))

1.2 Danbooru Reference

pybooru.danbooru

This module contains Danbooru class for access to API calls, authentication, build url and return JSON response.

Classes: Danbooru – Danbooru main class.

class pybooru.danbooru.**Danbooru** (*site_name=""*, *site_url=""*, *username=""*, *api_key=""*)
 Bases: *pybooru.pybooru._Pybooru*, *pybooru.api_danbooru.DanbooruApi_Mixin*

Danbooru class (inherits: Pybooru and DanbooruApi_Mixin).

To initialize Pybooru, you need to specify one of these two parameters: ‘site_name’ or ‘site_url’. If you specify ‘site_name’, Pybooru checks whether there is in the list of default sites (You can get list of sites in the ‘resources’ module).

To specify a site that isn’t in list of default sites, you need use ‘site_url’ parameter and specify url.

Some actions may require you to log in. always specify two parameters to log in: ‘username’ and ‘api_key’.

site_name

Get or set site name set.

Type str

site_url

Get or set the URL of Moebooru/Danbooru based site.

Type str

username

Return user name.

Type str

api_key

Return API key.

Type str

last_call

Return last call.

Type dict

__init__ (*site_name=""*, *site_url=""*, *username=""*, *api_key=""*)

Initialize Danbooru.

Keyword Arguments

- **site_name** (*str*) – Get or set site name set.
- **site_url** (*str*) – Get or set the URL of Moebooru/Danbooru based site.
- **username** (*str*) – Your username of the site (Required only for functions that modify the content).
- **api_key** (*str*) – Your api key of the site (Required only for functions that modify the content).

_get (*api_call*, *params=None*, *method='GET'*, *auth=False*, *file_=None*)
Function to preapre API call.

Parameters

- **api_call** (*str*) – API function to be called.
- **params** (*str*) – API function parameters.
- **method** (*str*) – (Default: GET) HTTP method (GET, POST, PUT or DELETE)
- **file** (*file*) – File to upload (only uploads).

Raise: PybooruError: When 'username' or 'api_key' are not set.

1.3 Danbooru API Reference

pybooru.api_danbooru

This module contains all API calls of Danbooru.

Classes: DanbooruApi_Mixin – Contains all API endpoints.

class pybooru.api_danbooru.DanbooruApi_Mixin
Bases: object

Contains all Danbooru API calls.

- API Version commit: 9996030
- Doc: https://danbooru.donmai.us/wiki_pages/43568

post_list (***params*)
Get a list of posts.

Parameters

- **limit** (*int*) – How many posts you want to retrieve. There is a hard limit of 100 posts per request.
- **page** (*int*) – The page number.
- **tags** (*str*) – The tags to search for. Any tag combination that works on the web site will work here. This includes all the meta-tags.
- **md5** (*str*) – The md5 of the image to search for.
- **random** (*bool*) – Can be: True, False.
- **raw** (*bool*) – When this parameter is set the tags parameter will not be parsed for aliased tags, metatags or multiple tags, and will instead be parsed as a single literal tag.

post_show (*post_id*)
Get a post.

Parameters `post_id (int)` – Where `post_id` is the post id.

post_update (`post_id`, `tag_string=None`, `rating=None`, `source=None`, `parent_id=None`,
`has_embedded_notes=None`, `is_rating_locked=None`, `is_note_locked=None`,
`is_status_locked=None`)

Update a specific post (Requires login).

Parameters

- **post_id** (`int`) – The id number of the post to update.
- **tag_string** (`str`) – A space delimited list of tags.
- **rating** (`str`) – The rating for the post. Can be: safe, questionable, or explicit.
- **source** (`str`) – If this is a URL, Danbooru will download the file.
- **parent_id** (`int`) – The ID of the parent post.
- **has_embedded_notes** (`int`) – Can be 1, 0.
- **is_rating_locked** (`int`) – Can be: 0, 1 (Builder+ only).
- **is_note_locked** (`int`) – Can be: 0, 1 (Builder+ only).
- **is_status_locked** (`int`) – Can be: 0, 1 (Admin only).

post_revert (`post_id`, `version_id`)

Function to reverts a post to a previous version (Requires login).

Parameters

- **post_id** (`int`) –
- **version_id** (`int`) – The post version id to revert to.

post_copy_notes (`post_id`, `other_post_id`)

Function to copy notes (requires login).

Parameters

- **post_id** (`int`) –
- **other_post_id** (`int`) – The id of the post to copy notes to.

post_mark_translated (`post_id`, `check_translation`, `partially_translated`)

Mark post as translated (Requires login) (UNTESTED).

If you set `check_translation` and `partially_translated` to 1 post will be tagged as ‘translated_request’

Parameters

- **post_id** (`int`) –
- **check_translation** (`int`) – Can be 0, 1.
- **partially_translated** (`int`) – Can be 0, 1

post_vote (`post_id`, `score`)

Action lets you vote for a post (Requires login). Danbooru: Post votes/create.

Parameters

- **post_id** (`int`) –
- **score** (`str`) – Can be: up, down.

post_unvote (`post_id`)

Action lets you unvote for a post (Requires login).

Parameters **post_id**(*int*) –

post_flag_list(*creator_id=None, creator_name=None, post_id=None, reason_matches=None, is_resolved=None, category=None*)

Function to flag a post (Requires login).

Parameters

- **creator_id**(*int*) – The user id of the flag’s creator.
- **creator_name**(*str*) – The name of the flag’s creator.
- **post_id**(*int*) – The post id if the flag.

post_flag_show(*flag_id*)

Show specific flagged post (Requires login).

Parameters **flag_id**(*int*) –

post_flag_create(*post_id, reason*)

Function to flag a post.

Parameters

- **post_id**(*int*) – The id of the flagged post.
- **reason**(*str*) – The reason of the flagging.

post_appeals_list(*creator_id=None, creator_name=None, post_id=None*)

Function to return list of appeals (Requires login).

Parameters

- **creator_id**(*int*) – The user id of the appeal’s creator.
- **creator_name**(*str*) – The name of the appeal’s creator.
- **post_id**(*int*) – The post id if the appeal.

post_appeals_show(*appeal_id*)

Show a specific post appeal (Requires login) (UNTESTED).

Parameters **appeal_id** –

post_appeals_create(*post_id, reason*)

Function to create appeals (Requires login).

Parameters

- **post_id**(*int*) – The id of the appealed post.
- **reason**(*str*) –

post_versions_list(*updater_name=None, updater_id=None, post_id=None, start_id=None*)

Get list of post versions.

Parameters

- **updater_name**(*str*) –
- **updater_id**(*int*) –
- **post_id**(*int*) –
- **start_id**(*int*) –

post_versions_show(*version_id*)

Show a specific post version (UNTESTED).

Parameters **version_id** (*int*) –

post_versions_undo (*version_id*)

Undo post version (Requires login) (UNTESTED).

Parameters **version_id** (*int*) –

count_posts (*tags=None*)

Show the number of posts on Danbooru or a specific tag search.

Parameters **tags** (*str*) –

upload_list (*uploader_id=None, uploader_name=None, source=None*)

Search and return an uploads list (Requires login).

Parameters

- **uploader_id** (*int*) – The id of the uploader.
- **uploader_name** (*str*) – The name of the uploader.
- **source** (*str*) – The source of the upload (exact string match).

upload_show (*upload_id*)

Get an upload (Requires login).

Parameters **upload_id** (*int*) –

upload_create (*tags, rating, file_=None, source=None, parent_id=None*)

Function to create a new upload (Requires login).

Parameters

- **tags** (*str*) –
- **rating** (*str*) – Can be: *s*, *q*, or *e*. Alternatively, you can specify *rating:safe*, *rating:questionable*, or *rating:explicit* in the tag string.
- **file** (*file_path*) – The file data encoded as a multipart form.
- **source** (*str*) – The source URL.
- **parent_id** (*int*) – The parent post id.

Raises `PybooruAPIError` – When **file_** or **source** are empty.

comment_list (*group_by, limit=None, page=None, body_matches=None, post_id=None, post_tags_match=None, creator_name=None, creator_id=None, is_deleted=None*)

Return a list of comments.

Parameters

- **limit** (*int*) – How many posts you want to retrieve.
- **page** (*int*) – The page number.
- **group_by** – Can be 'comment', 'post'. Comment will return recent comments. Post will return posts that have been recently commented on.
- **body_matches** (*str*) – Body contains the given terms.
- **post_id** (*int*) –
- **post_tags_match** (*str*) – The comment's post's tags match the given terms. Meta-tags not supported.
- **creator_name** (*str*) – The name of the creator (exact match).
- **creator_id** (*int*) – The user id of the creator.

- **is_deleted** (*bool*) – Can be: True, False.

Raises `PybooruAPIError` – When ‘group_by’ is invalid.

comment_create (*post_id, body, do_not_bump_post=None*)

Action to lets you create a comment (Requires login).

Parameters

- **post_id** (*int*) –
- **body** (*str*) –
- **do_not_bump_post** (*bool*) – Set to 1 if you do not want the post to be bumped to the top of the comment listing.

comment_update (*comment_id, body*)

Function to update a comment (Requires login).

Parameters

- **comment_id** (*int*) –
- **body** (*str*) –

comment_show (*comment_id*)

Get a specific comment.

Parameters **comment_id** (*int*) – The id number of the comment to retrieve.

comment_delete (*comment_id*)

Remove a specific comment (Requires login).

Parameters **comment_id** (*int*) – The id number of the comment to remove.

comment_undelete (*comment_id*)

Undelete a specific comment (Requires login) (UNTESTED).

Parameters **comment_id** (*int*) –

comment_vote (*comment_id, score*)

Lets you vote for a comment (Requires login).

Parameters

- **comment_id** (*int*) –
- **score** (*str*) – Can be: up, down.

comment_unvote (*comment_id*)

Lets you unvote a specific comment (Requires login).

Parameters **comment_id** (*int*) –

favorite_list (*user_id=None*)

Return a list with favorite posts (Requires login).

Parameters **user_id** (*int*) – Which user’s favorites to show. Defaults to your own if not specified.

favorite_add (*post_id*)

Add post to favorite (Requires login).

Parameters **post_id** (*int*) – The post to favorite.

favorite_remove (*post_id*)

Remove a post from favorites (Requires login).

Parameters `post_id (int)` – Where `post_id` is the post id.

dmail_list (`message_matches=None`, `to_name=None`, `to_id=None`, `from_name=None`, `from_id=None`, `read=None`)

Return list of Dmails. You can only view dmails you own (Requires login).

Parameters

- **message_matches** (`str`) – The message body contains the given terms.
- **to_name** (`str`) – The recipient's name.
- **to_id** (`int`) – The recipient's user id.
- **from_name** (`str`) – The sender's name.
- **from_id** (`int`) – The sender's user id.
- **read** (`bool`) – Can be: true, false.

dmail_show (`dmail_id`)

Return a specific dmail. You can only view dmails you own (Requires login).

Parameters `dmail_id (int)` – Where `dmail_id` is the dmail id.

dmail_create (`to_name`, `title`, `body`)

Create a dmail (Requires login)

Parameters

- **to_name** (`str`) – The recipient's name.
- **title** (`str`) – The title of the message.
- **body** (`str`) – The body of the message.

dmail_delete (`dmail_id`)

Delete a dmail. You can only delete dmails you own (Requires login).

Parameters `dmail_id (int)` – where `dmail_id` is the dmail id.

artist_list (`query=None`, `artist_id=None`, `creator_name=None`, `creator_id=None`, `is_active=None`, `is_banned=None`, `empty_only=None`, `order=None`)

Get an artist of a list of artists.

Parameters

- **query** (`str`) – This field has multiple uses depending on what the query starts with:
 'http:desired_url':
 Search for artist with this URL.

 'name:desired_url': Search for artists with the given name as their base name.
 'other:other_name': Search for artists with the given name in their other names.
 'group:group_name': Search for artists belonging to the group with the given name.
 'status:banned': Search for artists that are banned. else Search for the given name in the
 base name and the other names.
- **artist_id** (`id`) – The artist id.
- **creator_name** (`str`) – Exact creator name.
- **creator_id** (`id`) – Artist creator id.
- **is_active** (`bool`) – Can be: true, false

- **is_banned** (*bool*) – Can be: true, false
- **empty_only** (*True*) – Search for artists that have 0 posts. Can be: true
- **order** (*str*) – Can be: name, updated_at.

artist_show (*artist_id*)
Return a specific artist.

Parameters **artist_id** (*int*) – Where artist_id is the artist id.

artist_create (*name*, *other_names_comma=None*, *group_name=None*, *url_string=None*,
body=None)
Function to create an artist (Requires login) (UNTESTED).

Parameters

- **name** (*str*) –
- **other_names_comma** (*str*) – List of alternative names for this artist, comma delimited.
- **group_name** (*str*) – The name of the group this artist belongs to.
- **url_string** (*str*) – List of URLs associated with this artist, whitespace or newline delimited.
- **body** (*str*) – DText that will be used to create a wiki entry at the same time.

artist_update (*artist_id*, *name=None*, *other_names_comma=None*, *group_name=None*,
url_string=None, *body=None*)
Function to update artists (Requires login) (UNTESTED).

Parameters

- **artist_id** (*str*) –
- **name** (*str*) – Artist name.
- **other_names_comma** (*str*) – List of alternative names for this artist, comma delimited.
- **group_name** (*str*) – The name of the group this artist belongs to.
- **url_string** (*str*) – List of URLs associated with this artist, whitespace or newline delimited.
- **body** (*str*) – DText that will be used to create/update a wiki entry at the same time.

artist_delete (*artist_id*)
Action to lets you delete an artist (Requires login) (UNTESTED) (Only Builder+).

Parameters **artist_id** (*int*) – Where artist_id is the artist id.

artist_undelete (*artist_id*)
Lets you undelete artist (Requires login) (UNTESTED) (Only Builder+).

Parameters **artist_id** (*int*) –

artist_banned ()
This is a shortcut for an artist listing search with name=status:banned.

artist_revert (*artist_id*, *version_id*)
Revert an artist (Requires login) (UNTESTED).

Parameters

- **artist_id** (*int*) – The artist id.

- **version_id** (*int*) – The artist version id to revert to.

artist_versions (*name=None, updater_name=None, updater_id=None, artist_id=None, is_active=None, is_banned=None, order=None*)

Get list of artist versions (Requires login).

Parameters

- **name** (*str*) –
- **updater_name** (*str*) –
- **updater_id** (*int*) –
- **artist_id** (*int*) –
- **is_active** (*bool*) – Can be: True, False.
- **is_banned** (*bool*) – Can be: True, False.
- **order** (*str*) – Can be: name (Defaults to ID)

artist_commentary_list (*text_matches=None, post_id=None, post_tags_match=None, original_present=None, translated_present=None*)

list artist commentary.

Parameters

- **text_matches** (*str*) –
- **post_id** (*int*) –
- **post_tags_match** (*str*) – The commentary's post's tags match the giventerms. Meta-tags not supported.
- **original_present** (*str*) – Can be: yes, no.
- **translated_present** (*str*) – Can be: yes, no.

artist_commentary_create_update (*post_id, original_title, original_description, translated_title, translated_description*)

Create or update artist commentary (Requires login) (UNTESTED).

Parameters

- **post_id** (*int*) – Post id.
- **original_title** (*str*) – Original title.
- **original_description** (*str*) – Original description.
- **translated_title** (*str*) – Translated title.
- **translated_description** (*str*) – Translated description.

artist_commentary_revert (*id_, version_id*)

Revert artist commentary (Requires login) (UNTESTED).

Parameters

- **id** (*int*) – The artist commentary id.
- **version_id** (*int*) – The artist commentary version id to revert to.

artist_commentary_versions (*post_id, updater_id*)

Return list of artist commentary versions.

Parameters

- **updater_id**(*int*) –

- **post_id**(*int*) –

note_list(*body_matches=None, post_id=None, post_tags_match=None, creator_name=None, creator_id=None, is_active=None*)

Return list of notes.

Parameters

- **body_matches**(*str*) – The note's body matches the given terms.

- **post_id**(*int*) – A specific post.

- **post_tags_match**(*str*) – The note's post's tags match the given terms.

- **creator_name**(*str*) – The creator's name. Exact match.

- **creator_id**(*int*) – The creator's user id.

- **is_active**(*bool*) – Can be: True, False.

note_show(*note_id*)

Get a specific note.

Parameters **note_id**(*int*) – Where note_id is the note id.

note_create(*post_id, coor_x, coor_y, width, height, body*)

Function to create a note (Requires login) (UNTESTED).

Parameters

- **post_id**(*int*) –

- **coor_x**(*int*) – The x coordinates of the note in pixels, with respect to the top-left corner of the image.

- **coor_y**(*int*) – The y coordinates of the note in pixels, with respect to the top-left corner of the image.

- **width**(*int*) – The width of the note in pixels.

- **height**(*int*) – The height of the note in pixels.

- **body**(*str*) – The body of the note.

note_update(*note_id, coor_x=None, coor_y=None, width=None, height=None, body=None*)

Function to update a note (Requires login) (UNTESTED).

Parameters

- **note_id**(*int*) – Where note_id is the note id.

- **coor_x**(*int*) – The x coordinates of the note in pixels, with respect to the top-left corner of the image.

- **coor_y**(*int*) – The y coordinates of the note in pixels, with respect to the top-left corner of the image.

- **width**(*int*) – The width of the note in pixels.

- **height**(*int*) – The height of the note in pixels.

- **body**(*str*) – The body of the note.

note_delete(*note_id*)

delete a specific note (Requires login) (UNTESTED).

Parameters **note_id**(*int*) – Where note_id is the note id.

note_revert (*note_id*, *version_id*)

Function to revert a specific note (Requires login) (UNTESTED).

Parameters

- **note_id** (*int*) – Where note_id is the note id.
- **version_id** (*int*) – The note version id to revert to.

note_versions (*updater_id=None*, *post_id=None*, *note_id=None*)

Get list of note versions.

Parameters

- **updater_id** (*int*) –
- **post_id** (*int*) –
- **note_id** (*int*) –

user_list (*name=None*, *name_matches=None*, *min_level=None*, *max_level=None*, *level=None*, *user_id=None*, *order=None*)

Function to get a list of users or a specific user.

Levels: Users have a number attribute called level representing their role. The current levels are:

Member 20, Gold 30, Platinum 31, Builder 32, Contributor 33, Janitor 35, Moderator 40 and Admin 50.

Parameters

- **name** (*str*) – Supports patterns.
- **name_matches** (*str*) – Same functionality as name.
- **min_level** (*int*) – Minimum level (see section on levels).
- **max_level** (*int*) – Maximum level (see section on levels).
- **level** (*int*) – Current level (see section on levels).
- **user_id** (*int*) – The user id.
- **order** (*str*) – Can be: 'name', 'post_upload_count', 'note_count', 'post_update_count', 'date'.

user_show (*user_id*)

Get a specific user.

Parameters **user_id** (*int*) – Where user_id is the user id.

pool_list (*name_matches=None*, *pool_ids=None*, *category=None*, *description_matches=None*, *creator_name=None*, *creator_id=None*, *is_deleted=None*, *is_active=None*, *order=None*)

Get a list of pools.

Parameters

- **name_matches** (*str*) –
- **pool_ids** (*str*) – Can search for multiple ID's at once, separated by commas.
- **description_matches** (*str*) –
- **creator_name** (*str*) –
- **creator_id** (*int*) –
- **is_active** (*bool*) – Can be: true, false.

- **is_deleted** (*bool*) – Can be: True, False.
- **order** (*str*) – Can be: name, created_at, post_count, date.
- **category** (*str*) – Can be: series, collection.

pool_show (*pool_id*)

Get a specific pool.

Parameters **pool_id** (*int*) – Where pool_id is the pool id.

pool_create (*name, description, category*)

Function to create a pool (Requires login) (UNTESTED).

Parameters

- **name** (*str*) – Pool name.
- **description** (*str*) – Pool description.
- **category** (*str*) – Can be: series, collection.

pool_update (*pool_id, name=None, description=None, post_ids=None, is_active=None, category=None*)

Update a pool (Requires login) (UNTESTED).

Parameters

- **pool_id** (*int*) – Where pool_id is the pool id.
- **name** (*str*) –
- **description** (*str*) –
- **post_ids** (*str*) – List of space delimited post ids.
- **is_active** (*int*) – Can be: 1, 0.
- **category** (*str*) – Can be: series, collection.

pool_delete (*pool_id*)

Delete a pool (Requires login) (UNTESTED) (Moderator+).

Parameters **pool_id** (*int*) – Where pool_id is the pool id.

pool_undelete (*pool_id*)

Undelete a specific pool (Requires login) (UNTESTED) (Moderator+).

Parameters **pool_id** (*int*) – Where pool_id is the pool id.

pool_revert (*pool_id, version_id*)

Function to revert a specific pool (Requires login) (UNTESTED).

Parameters

- **pool_id** (*int*) – Where pool_id is the pool id.
- **version_id** (*int*) –

pool_versions (*updater_id=None, updater_name=None, pool_id=None*)

Get list of pool versions.

Parameters

- **updater_id** (*int*) –
- **updater_name** (*str*) –
- **pool_id** (*int*) –

tag_list (*name_matches=None, name=None, category=None, hide_empty=None, has_wiki=None, has_artist=None, order=None*)
Get a list of tags.

Parameters

- **name_matches** (*str*) – Can be: part or full name.
- **name** (*str*) – Allows searching for multiple tags with exact given names, separated by commas. e.g. search[name]=touhou,original,k-on! would return the three listed tags.
- **category** (*str*) – Can be: 0, 1, 3, 4 (general, artist, copyright, character respectively).
- **hide_empty** (*str*) – Can be: yes, no. Excludes tags with 0 posts when “yes”.
- **has_wiki** (*str*) – Can be: yes, no.
- **has_artist** (*str*) – Can be: yes, no.
- **order** (*str*) – Can be: name, date, count.

tag_show (*tag_id*)
Show a specific tag.

Parameters **tag_id** (*int*) –

tag_update (*tag_id, category*)
Lets you update a tag (Requires login) (UNTESTED).

Parameters

- **tag_id** (*int*) –
- **category** (*str*) – Can be: 0, 1, 3, 4 (general, artist, copyright, character respectively).

tag_aliases (*name_matches=None, antecedent_name=None, tag_id=None*)
Get tags aliases.

Parameters

- **name_matches** (*str*) – Match antecedent or consequent name.
- **antecedent_name** (*str*) – Match antecedent name (exact match).
- **tag_id** (*int*) – The tag alias id.

tag_implications (*name_matches=None, antecedent_name=None, tag_id=None*)
Get tags implications.

Parameters

- **name_matches** (*str*) – Match antecedent or consequent name.
- **antecedent_name** (*str*) – Match antecedent name (exact match).
- **tag_id** (*int*) – Tag implication id.

tag_related (*query, category=None*)
Get related tags.

Parameters

- **query** (*str*) – The tag to find the related tags for.
- **category** (*str*) – If specified, show only tags of a specific category. Can be: General 0, Artist 1, Copyright 3 and Character 4.

wiki_list (*title=None, creator_id=None, body_matches=None, other_names_match=None, creator_name=None, hide_deleted=None, other_names_present=None, order=None*)

Function to retrieves a list of every wiki page.

Parameters

- **title** (*str*) – Page title.
- **creator_id** (*int*) – Creator id.
- **body_matches** (*str*) – Page content.
- **other_names_match** (*str*) – Other names.
- **creator_name** (*str*) – Creator name.
- **hide_deleted** (*str*) – Can be: yes, no.
- **other_names_present** (*str*) – Can be: yes, no.
- **order** (*str*) – Can be: date, title.

wiki_show (*wiki_page_id*)

Retrieve a specific page of the wiki.

Parameters **wiki_page_id** (*int*) – Where page_id is the wiki page id.

wiki_create (*title, body, other_names=None*)

Action to lets you create a wiki page (Requires login) (UNTESTED).

Parameters

- **title** (*str*) – Page title.
- **body** (*str*) – Page content.
- **other_names** (*str*) – Other names.

wiki_update (*page_id, title=None, body=None, other_names=None, is_locked=None, is_deleted=None*)

Action to lets you update a wiki page (Requires login) (UNTESTED).

Parameters

- **page_id** (*int*) – Whre page_id is the wiki page id.
- **title** (*str*) – Page title.
- **body** (*str*) – Page content.
- **other_names** (*str*) – Other names.
- **is_locked** (*int*) – Can be: 0, 1 (Builder+).
- **is_deleted** (*int*) – Can be: 0, 1 (Builder+).

wiki_delete (*page_id*)

Delete a specific page wiki (Requires login) (UNTESTED) (Builder+).

Parameters **page_id** (*int*) –

wiki_revert (*wiki_page_id, version_id*)

Revert page to a previous version (Requires login) (UNTESTED).

Parameters

- **wiki_page_id** (*int*) – Where page_id is the wiki page id.
- **version_id** (*int*) –

wiki_versions_list (*page_id*, *updater_id*)

Return a list of wiki page version.

Parameters

- **page_id** (*int*) –
- **updater_id** (*int*) –

wiki_versions_show (*page_id*)

Return a specific wiki page version.

Parameters **page_id** (*int*) – Where *page_id* is the wiki page version id.

forum_topic_list (*title_matches=None*, *title=None*, *category_id=None*)

Function to get forum topics.

Parameters

- **title_matches** (*str*) – Search body for the given terms.
- **title** (*str*) – Exact title match.
- **category_id** (*int*) – Can be: 0, 1, 2 (General, Tags, Bugs & Features respectively).

forum_topic_show (*topic_id*)

Retrieve a specific forum topic.

Parameters **topic_id** (*int*) – Where *topic_id* is the forum topic id.

forum_topic_create (*title*, *body*, *category=None*)

Function to create topic (Requires login) (UNTESTED).

Parameters

- **title** (*str*) – topic title.
- **body** (*str*) – Message of the initial post.
- **category** (*str*) – Can be: 0, 1, 2 (General, Tags, Bugs & Features respectively).

forum_topic_update (*topic_id*, *title=None*, *category=None*)

Update a specific topic (Login Requires) (UNTESTED).

Parameters

- **topic_id** (*int*) – Where *topic_id* is the topic id.
- **title** (*str*) – Topic title.
- **category** (*str*) – Can be: 0, 1, 2 (General, Tags, Bugs & Features respectively).

forum_topic_delete (*topic_id*)

Delete a topic (Login Requires) (Moderator+) (UNTESTED).

Parameters **topic_id** (*int*) – Where *topic_id* is the topic id.

forum_topic_undelete (*topic_id*)

Un delete a topic (Login requires) (Moderator+) (UNTESTED).

Parameters **topic_id** (*int*) – Where *topic_id* is the topic id.

forum_post_list (*creator_id=None*, *creator_name=None*, *topic_id=None*,
topic_title_matches=None, *topic_category_id=None*, *body_matches=None*)

Return a list of forum posts.

Parameters

- **creator_id** (*int*) –

- **creator_name** (*str*) –
- **topic_id** (*int*) –
- **topic_title_matches** (*str*) –
- **topic_category_id** (*int*) – Can be: 0, 1, 2 (General, Tags, Bugs & Features respectively).
- **body_matches** (*str*) – Can be part of the post content.

forum_post_create (*topic_id*, *body*)
Create a forum post (Requires login).

Parameters

- **topic_id** (*int*) –
- **body** (*str*) – Post content.

forum_post_update (*topic_id*, *body*)
Update a specific forum post (Requires login)(Moderator+)(UNTESTED).

Parameters

- **post_id** (*int*) – Forum topic id.
- **body** (*str*) – Post content.

forum_post_delete (*post_id*)
Delete a specific forum post (Requires login)(Moderator+)(UNTESTED).

Parameters **post_id** (*int*) – Forum post id.

forum_post_undelete (*post_id*)
Undelete a specific forum post (Requires login)(Moderator+)(UNTESTED).

Parameters **post_id** (*int*) – Forum post id.

1.4 Moebooru Reference

pybooru.moebooru

This module contains Moebooru class for access to API calls, authentication, build url and return JSON response.

Classes: Moebooru – Moebooru classes.

class pybooru.moebooru.**Moebooru** (*site_name*=", *site_url*", *username*", *password*",
hash_string", *api_version*='1.13.0+update.3')

Bases: [pybooru.pybooru._Pybooru](#), [pybooru.api_moebooru.MoebooruApi_Mixin](#)

Moebooru class (inherits: Pybooru and MoebooruApi_Mixin).

To initialize Pybooru, you need to specify one of these two parameters: 'site_name' or 'site_url'. If you specify 'site_name', Pybooru checks whether there is in the list of default sites (You can get list of sites in the 'resources' module).

To specify a site that isn't in list of default sites, you need use 'site_url' parameter and specify url.

Some actions may require you to log in. always specify three parameters to log in: 'hash_string', 'username' and 'password'. Default sites has an associate hash string.

site_name

Get or set site name set.

Type str

site_url

Get or set the URL of Moebooru/Danbooru based site.

Type str

api_version

Version of Moebooru API.

Type str

username

Return user name.

Type str

password

Return password in plain text.

Type str

hash_string

Return hash_string of the site.

Type str

last_call

Type dict

__init__ (*site_name*=", *site_url*=", *username*=", *password*=", *hash_string*=",
api_version='1.13.0+update.3')
 Initialize Moebooru.

Keyword Arguments

- **site_name** (*str*) – Get or set site name set.
- **site_url** (*str*) – Get or set the URL of Moebooru/Danbooru based site.
- **api_version** (*str*) – Version of Moebooru API.
- **hash_string** (*str*) – String that is hashed (required to login). (See the API documentation of the site for more information).
- **username** (*str*) – Your username of the site (Required only for functions that modify the content).
- **password** (*str*) – Your user password in plain text (Required only for functions that modify the content).

site_name

Get or set site name.

Getter Return site name.

Setter Validate and set site name.

Type string

_build_url (*api_call*)

Build request url.

Parameters **api_call** (*str*) – Base API Call.

Returns Complete url (str).

`_build_hash_string()`

Function for build password hash string.

Raises

- `PybooruError` – When isn't provide hash string.
- `PybooruError` – When aren't provide username or password.
- `PybooruError` – When Pybooru can't add password to hash string.

`_get(api_call, params, method='GET', file_=None)`

Function to preapre API call.

Parameters

- **`api_call`** (*str*) – API function to be called.
- **`params`** (*dict*) – API function parameters.
- **`method`** (*str*) – (Default: GET) HTTP method 'GET' or 'POST'
- **`file`** (*file*) – File to upload.

1.5 Moebooru API Reference

`pybooru.api_moebooru`

This module contains all API calls of Moebooru.

Classes: `MoebooruApi_Mixin` – Contains all API calls.

class `pybooru.api_moebooru.MoebooruApi_Mixin`

Bases: `object`

Contains all Moebooru API calls.

- API Versions: 1.13.0+update.3 and 1.13.0
- doc: <https://yande.re/help/api> or <https://konachan.com/help/api>

`post_list(**params)`

Get a list of posts.

Parameters

- **`tags`** (*str*) – The tags to search for. Any tag combination that works on the web site will work here. This includes all the meta-tags.
- **`limit`** (*int*) – How many posts you want to retrieve. There is a limit of 100:param posts per request.
- **`page`** (*int*) – The page number.

`post_create(tags, file_=None, rating=None, source=None, rating_locked=None, note_locked=None, parent_id=None, md5=None)`

Function to create a new post (Requires login).

There are only two mandatory fields: you need to supply the 'tags', and you need to supply the '**file_**', either through a multipart form or through a source URL (Requires login) (UNTESTED).

Parameters

- **`tags`** (*str*) – A space delimited list of tags.

- **file** (*str*) – The file data encoded as a multipart form. Path of content.
- **rating** (*str*) – The rating for the post. Can be: safe, questionable, or explicit.
- **source** (*str*) – If this is a URL, Moebooru will download the file.
- **rating_locked** (*bool*) – Set to True to prevent others from changing the rating.
- **note_locked** (*bool*) – Set to True to prevent others from adding notes.
- **parent_id** (*int*) – The ID of the parent post.
- **md5** (*str*) – Supply an MD5 if you want Moebooru to verify the file after uploading. If the MD5 doesn't match, the post is destroyed.

Raises `PybooruAPIError` – When file or source are empty.

post_update (*post_id*, *tags=None*, *file_=None*, *rating=None*, *source=None*, *is_rating_locked=None*, *is_note_locked=None*, *parent_id=None*)

Update a specific post.

Only the 'post_id' parameter is required. Leave the other parameters blank if you don't want to change them (Requires login).

Parameters

- **post_id** (*int*) – The id number of the post to update.
- **tags** (*str*) – A space delimited list of tags. Specify previous tags.
- **file** (*str*) – The file data ENCODED as a multipart form.
- **rating** (*str*) – The rating for the post. Can be: safe, questionable, or explicit.
- **source** (*str*) – If this is a URL, Moebooru will download the file.
- **rating_locked** (*bool*) – Set to True to prevent others from changing the rating.
- **note_locked** (*bool*) – Set to True to prevent others from adding notes.
- **parent_id** (*int*) – The ID of the parent post.

post_destroy (*post_id*)

Function to destroy a specific post.

You must also be the user who uploaded the post (or you must be a moderator) (Requires Login) (UNTESTED).

Parameters **post_id** (*int*) – The id number of the post to delete.

post_revert_tags (*post_id*, *history_id*)

Function to reverts a post to a previous set of tags (Requires login) (UNTESTED).

Parameters

- **post_id** (*int*) – The post id number to update.
- **history_id** (*int*) – The id number of the tag history.

post_vote (*post_id*, *score*)

Action lets you vote for a post (Requires login).

Parameters

- **post_id** (*int*) – The post id.
- **score** (*int*) –
– 0: No voted or Remove vote.

- 1: Good.
- 2: Great.
- 3: Favorite, add post to favorites.

Raises `PybooruAPIError` – When score is > 3.

tag_list (***params*)
Get a list of tags.

Parameters

- **name** (*str*) – The exact name of the tag.
- **id** (*int*) – The id number of the tag.
- **limit** (*int*) – How many tags to retrieve. Setting this to 0 will return every tag (Default value: 0).
- **page** (*int*) – The page number.
- **order** (*str*) – Can be 'date', 'name' or 'count'.
- **after_id** (*int*) – Return all tags that have an id number greater than this.

tag_update (*name=None, tag_type=None, is_ambiguous=None*)
Action to lets you update tag (Requires login) (UNTESTED).

Parameters

- **name** (*str*) – The name of the tag to update.
- **tag_type** (*int*) –
 - General: 0.
 - artist: 1.
 - copyright: 3.
 - character: 4.
- **is_ambiguous** (*int*) – Whether or not this tag is ambiguous. Use 1 for True and 0 for False.

tag_related (***params*)
Get a list of related tags.

Parameters

- **tags** (*str*) – The tag names to query.
- **type** (*str*) – Restrict results to this tag type. Can be general, artist, copyright, or character.

artist_list (***params*)
Get a list of artists.

Parameters

- **name** (*str*) – The name (or a fragment of the name) of the artist.
- **order** (*str*) – Can be date or name.
- **page** (*int*) – The page number.

artist_create (*name, urls=None, alias=None, group=None*)
Function to create an artist (Requires login) (UNTESTED).

Parameters

- **name** (*str*) – The artist’s name.
- **urls** (*str*) – A list of URLs associated with the artist, whitespace delimited.
- **alias** (*str*) – The artist that this artist is an alias for. Simply enter the alias artist’s name.
- **group** (*str*) – The group or cicle that this artist is a member of. Simply:param enter the group’s name.

artist_update (*artist_id*, *name=None*, *urls=None*, *alias=None*, *group=None*)

Function to update artists (Requires Login) (UNTESTED).

Only the artist_id parameter is required. The other parameters are optional.

Parameters

- **artist_id** (*int*) – The id of thr artist to update (Type: INT).
- **name** (*str*) – The artist’s name.
- **urls** (*str*) – A list of URLs associated with the artist, whitespace delimited.
- **alias** (*str*) – The artist that this artist is an alias for. Simply enter the alias artist’s name.
- **group** (*str*) – The group or cicle that this artist is a member of. Simply enter the group’s name.

artist_destroy (*artist_id*)

Action to lets you remove artist (Requires login) (UNTESTED).

Parameters **artist_id** (*int*) – The id of the artist to destroy.

comment_show (*comment_id*)

Get a specific comment.

Parameters **comment_id** (*str*) – The id number of the comment to retrieve.

comment_create (*post_id*, *comment_body*, *anonymous=None*)

Action to lets you create a comment (Requires login).

Parameters

- **post_id** (*int*) – The post id number to which you are responding.
- **comment_body** (*str*) – The body of the comment.
- **anonymous** (*int*) – Set to 1 if you want to post this comment anonymously.

comment_destroy (*comment_id*)

Remove a specific comment (Requires login).

Parameters **comment_id** (*int*) – The id number of the comment to remove.

wiki_list (***params*)

Function to retrieves a list of every wiki page.

Parameters

- **query** (*str*) – A word or phrase to search for (Default: None).
- **order** (*str*) – Can be: title, date (Default: title).
- **limit** (*int*) – The number of pages to retrieve (Default: 100).

- **page** (*int*) – The page number.

wiki_create (*title, body*)

Action to lets you create a wiki page (Requires login) (UNTESTED).

Parameters

- **title** (*str*) – The title of the wiki page.
- **body** (*str*) – The body of the wiki page.

wiki_update (*title, new_title=None, page_body=None*)

Action to lets you update a wiki page (Requires login) (UNTESTED).

Parameters

- **title** (*str*) – The title of the wiki page to update.
- **new_title** (*str*) – The new title of the wiki page.
- **page_body** (*str*) – The new body of the wiki page.

wiki_show (***params*)

Get a specific wiki page.

Parameters

- **title** (*str*) – The title of the wiki page to retrieve.
- **version** (*int*) – The version of the page to retrieve.

wiki_destroy (*title*)

Function to delete a specific wiki page (Requires login) (Only moderators) (UNTESTED).

Parameters title (*str*) – The title of the page to delete.

wiki_lock (*title*)

Function to lock a specific wiki page (Requires login) (Only moderators) (UNTESTED).

Parameters title (*str*) – The title of the page to lock.

wiki_unlock (*title*)

Function to unlock a specific wiki page (Requires login) (Only moderators) (UNTESTED).

Parameters title (*str*) – The title of the page to unlock.

wiki_revert (*title, version*)

Function to revert a specific wiki page (Requires login) (UNTESTED).

Parameters

- **title** (*str*) – The title of the wiki page to update.
- **version** (*int*) – The version to revert to.

wiki_history (*title*)

Get history of specific wiki page.

Parameters title (*str*) – The title of the wiki page to retrieve versions for.

note_list (***params*)

Get note list.

Parameters post_id (*int*) – The post id number to retrieve notes for.

note_search (*query*)

Search specific note.

Parameters `query` (*str*) – A word or phrase to search for.

note_history (***params*)

Get history of notes.

Parameters

- **post_id** (*int*) – The post id number to retrieve note versions for.
- **id** (*int*) – The note id number to retrieve versions for.
- **limit** (*int*) – How many versions to retrieve (Default: 10).
- **page** (*int*) – The note id number to retrieve versions for.

note_revert (*note_id, version*)

Function to revert a specific note (Requires login) (UNTESTED).

Parameters

- **note_id** (*int*) – The note id to update.
- **version** (*int*) – The version to revert to.

note_create_update (*post_id=None, coor_x=None, coor_y=None, width=None, height=None, is_active=None, body=None, note_id=None*)

Function to create or update note (Requires login) (UNTESTED).

Parameters

- **post_id** (*int*) – The post id number this note belongs to.
- **coor_x** (*int*) – The X coordinate of the note.
- **coor_y** (*int*) – The Y coordinate of the note.
- **width** (*int*) – The width of the note.
- **height** (*int*) – The height of the note.
- **is_active** (*int*) – Whether or not the note is visible. Set to 1 for active, 0 for inactive.
- **body** (*str*) – The note message.
- **note_id** (*int*) – If you are updating a note, this is the note id number to update.

user_search (***params*)

Search users.

If you don't specify any parameters you'll _get a listing of all users.

Parameters

- **id** (*int*) – The id number of the user.
- **name** (*str*) – The name of the user.

forum_list (***params*)

Function to get forum posts.

If you don't specify any parameters you'll _get a listing of all users.

Parameters **parent_id** (*int*) – The parent ID number. You'll return all the responses to that forum post.

pool_list (***params*)

Function to get pools.

If you don't specify any parameters you'll get a list of all pools.

Parameters

- **query** (*str*) – The title.
- **page** (*int*) – The page number.

pool_posts (***params*)

Function to get pools posts.

If you don't specify any parameters you'll get a list of all pools.

Parameters

- **id** (*int*) – The pool id number.
- **page** (*int*) – The page number.

pool_update (*pool_id, name=None, is_public=None, description=None*)

Function to update a pool (Requires login) (UNTESTED).

Parameters

- **pool_id** (*int*) – The pool id number.
- **name** (*str*) – The name.
- **is_public** (*int*) – 1 or 0, whether or not the pool is public.
- **description** (*str*) – A description of the pool.

pool_create (*name, description, is_public*)

Function to create a pool (Require login) (UNTESTED).

Parameters

- **name** (*str*) – The name.
- **description** (*str*) – A description of the pool.
- **is_public** (*int*) – 1 or 0, whether or not the pool is public.

pool_destroy (*pool_id*)

Function to destroy a specific pool (Require login) (UNTESTED).

Parameters **pool_id** (*int*) – The pool id number.**pool_add_post** (***params*)

Function to add a post (Require login) (UNTESTED).

Parameters

- **pool_id** (*int*) – The pool to add the post to.
- **post_id** (*int*) – The post to add.

pool_remove_post (***params*)

Function to remove a post (Require login) (UNTESTED).

Parameters

- **pool_id** (*int*) – The pool to remove the post to.
- **post_id** (*int*) – The post to remove.

favorite_list_users (*post_id*)

Function to return a list with all users who have added to favorites a specific post.

Parameters **post_id** (*int*) – The post id.

1.6 Pybooru Reference

1.6.1 Pybooru

pybooru.pybooru

This module contains pybooru main class for access to API calls, authentication and return JSON response.

Classes: `_Pybooru` – Main pybooru classs, define Pybooru object and do requests.

class `pybooru.pybooru._Pybooru` (*site_name=""*, *site_url=""*, *username=""*)

Bases: `object`

Pybooru main class.

site_name

Get or set site name set.

Type `str`

site_url

Get or set the URL of Moebooru/Danbooru based site.

Type `str`

username

Return user name.

Type `str`

last_call

Return last call.

Type `dict`

__init__ (*site_name=""*, *site_url=""*, *username=""*)

Initialize Pybooru.

Keyword Arguments

- **site_name** (*str*) – The site name in ‘SITE_LIST’, default sites.
- **site_url** (*str*) – URL of on Moebooru/Danbooru based sites.
- **username** (*str*) – Your username of the site (Required only for functions that modify the content).

Raises `PybooruError` – When ‘site_name’ and ‘site_url’ are empty.

site_name

Get or set site name.

Getter Return site name.

Setter Validate and set site name.

Type `string`

__weakref__

list of weak references to the object (if defined)

site_url

Get or set site url.

Getter Return site url.

Setter Validate and set site url.

Type string

static `_get_status(status_code)`

Get status message for status code.

Parameters `status_code(int)` – HTTP status code.

Returns status message (str).

_request (`url, api_call, request_args, method='GET'`)

Function to request and returning JSON data.

Parameters

- `url(str)` – Base url call.
- `api_call(str)` – API function to be called.
- `request_args(dict)` – All requests parameters.
- `method(str)` – (Default: GET) HTTP method 'GET' or 'POST'

Raises

- `PybooruHTTPError` – HTTP Error.
- `requests.exceptions.Timeout` – When HTTP Timeout.
- `ValueError` – When can't decode JSON response.

1.6.2 Exceptions

`pybooru.exceptions`

This module contains Pybooru exceptions.

Classes:

- `PybooruError` – Main Pybooru exception class.
- `PybooruHTTPError` – Manages HTTP status errors.
- `PybooruAPIError` – Manages all API errors.

exception `pybooru.exceptions.PybooruError`

Bases: `Exception`

Class to catch Pybooru error message.

__weakref__

list of weak references to the object (if defined)

exception `pybooru.exceptions.PybooruHTTPError(msg, http_code, url)`

Bases: `pybooru.exceptions.PybooruError`

Class to catch HTTP error message.

__init__ (`msg, http_code, url`)

Initialize `PybooruHTTPError`.

Keyword Arguments

- `msg(str)` – The error message.
- `http_code(int)` – The HTTP status code.

- **url** (*str*) – The URL.

__str__ ()

Print exception.

exception pybooru.exceptions.**PybooruAPIError**

Bases: *pybooru.exceptions.PybooruError*

Class to catch all API errors.

1.6.3 Resources

pybooru.resources

This module contains all resources for Pybooru.

SITE_LIST (dict): Contains various Moebooru and Danbooru-based default sites.

HTTP_STATUS_CODE (dict): Contains http status codes for Moebooru and Danbooru API.

CHAPTER 2

Changelog

Changelog

p

- `pybooru.api_danbooru`, 6
- `pybooru.api_moebooru`, 22
- `pybooru.danbooru`, 5
- `pybooru.exceptions`, 30
- `pybooru.moebooru`, 20
- `pybooru.pybooru`, 29
- `pybooru.resources`, 31

Symbols

[_Pybooru \(class in pybooru.pybooru\)](#), 29
[__init__ \(\) \(pybooru.danbooru.Danbooru method\)](#), 5
[__init__ \(\) \(pybooru.exceptions.PybooruHTTPError method\)](#), 30
[__init__ \(\) \(pybooru.moebooru.Moebooru method\)](#), 21
[__init__ \(\) \(pybooru.pybooru._Pybooru method\)](#), 29
[__str__ \(\) \(pybooru.exceptions.PybooruHTTPError method\)](#), 31
[__weakref__ \(pybooru.exceptions.PybooruError attribute\)](#), 30
[__weakref__ \(pybooru.pybooru._Pybooru attribute\)](#), 29
[_build_hash_string \(\) \(pybooru.moebooru.Moebooru method\)](#), 21
[_build_url \(\) \(pybooru.moebooru.Moebooru method\)](#), 21
[_get \(\) \(pybooru.danbooru.Danbooru method\)](#), 6
[_get \(\) \(pybooru.moebooru.Moebooru method\)](#), 22
[_get_status \(\) \(pybooru.pybooru._Pybooru static method\)](#), 30
[_request \(\) \(pybooru.pybooru._Pybooru method\)](#), 30

A

[api_key \(pybooru.danbooru.Danbooru attribute\)](#), 5
[api_version \(pybooru.moebooru.Moebooru attribute\)](#), 21
[artist_banned \(\) \(pybooru.api_danbooru.DanbooruApi_Mixin method\)](#), 12
[artist_commentary_create_update \(\) \(pybooru.api_danbooru.DanbooruApi_Mixin method\)](#), 13
[artist_commentary_list \(\) \(pybooru.api_danbooru.DanbooruApi_Mixin method\)](#), 13
[artist_commentary_revert \(\) \(pybooru.api_danbooru.DanbooruApi_Mixin method\)](#), 13

[method\)](#), 13
[artist_commentary_versions \(\) \(pybooru.api_danbooru.DanbooruApi_Mixin method\)](#), 13
[artist_create \(\) \(pybooru.api_danbooru.DanbooruApi_Mixin method\)](#), 12
[artist_create \(\) \(pybooru.api_moebooru.MoebooruApi_Mixin method\)](#), 24
[artist_delete \(\) \(pybooru.api_danbooru.DanbooruApi_Mixin method\)](#), 12
[artist_destroy \(\) \(pybooru.api_moebooru.MoebooruApi_Mixin method\)](#), 25
[artist_list \(\) \(pybooru.api_danbooru.DanbooruApi_Mixin method\)](#), 11
[artist_list \(\) \(pybooru.api_moebooru.MoebooruApi_Mixin method\)](#), 24
[artist_revert \(\) \(pybooru.api_danbooru.DanbooruApi_Mixin method\)](#), 12
[artist_show \(\) \(pybooru.api_danbooru.DanbooruApi_Mixin method\)](#), 12
[artist_undelete \(\) \(pybooru.api_danbooru.DanbooruApi_Mixin method\)](#), 12
[artist_update \(\) \(pybooru.api_danbooru.DanbooruApi_Mixin method\)](#), 12
[artist_update \(\) \(pybooru.api_moebooru.MoebooruApi_Mixin method\)](#), 25
[artist_versions \(\) \(pybooru.api_danbooru.DanbooruApi_Mixin method\)](#), 13

C

`comment_create()` (py-booru.api_danbooru.DanbooruApi_Mixin method), 10

`comment_create()` (py-booru.api_moebooru.MoebooruApi_Mixin method), 25

`comment_delete()` (py-booru.api_danbooru.DanbooruApi_Mixin method), 10

`comment_destroy()` (py-booru.api_moebooru.MoebooruApi_Mixin method), 25

`comment_list()` (py-booru.api_danbooru.DanbooruApi_Mixin method), 9

`comment_show()` (py-booru.api_danbooru.DanbooruApi_Mixin method), 10

`comment_show()` (py-booru.api_moebooru.MoebooruApi_Mixin method), 25

`comment_undelete()` (py-booru.api_danbooru.DanbooruApi_Mixin method), 10

`comment_unvote()` (py-booru.api_danbooru.DanbooruApi_Mixin method), 10

`comment_update()` (py-booru.api_danbooru.DanbooruApi_Mixin method), 10

`comment_vote()` (py-booru.api_danbooru.DanbooruApi_Mixin method), 10

`count_posts()` (py-booru.api_danbooru.DanbooruApi_Mixin method), 9

D

`Danbooru` (class in `pybooru.danbooru`), 5

`DanbooruApi_Mixin` (class in `py-booru.api_danbooru`), 6

`dmail_create()` (py-booru.api_danbooru.DanbooruApi_Mixin method), 11

`dmail_delete()` (py-booru.api_danbooru.DanbooruApi_Mixin method), 11

`dmail_list()` (`pybooru.api_danbooru.DanbooruApi_Mixin` method), 11

`dmail_show()` (`pybooru.api_danbooru.DanbooruApi_Mixin` method), 11

F

`favorite_add()` (py-booru.api_danbooru.DanbooruApi_Mixin method), 10

`favorite_list()` (py-booru.api_danbooru.DanbooruApi_Mixin method), 10

`favorite_list_users()` (py-booru.api_moebooru.MoebooruApi_Mixin method), 28

`favorite_remove()` (py-booru.api_danbooru.DanbooruApi_Mixin method), 10

`forum_list()` (`pybooru.api_moebooru.MoebooruApi_Mixin` method), 27

`forum_post_create()` (py-booru.api_danbooru.DanbooruApi_Mixin method), 20

`forum_post_delete()` (py-booru.api_danbooru.DanbooruApi_Mixin method), 20

`forum_post_list()` (py-booru.api_danbooru.DanbooruApi_Mixin method), 19

`forum_post_undelete()` (py-booru.api_danbooru.DanbooruApi_Mixin method), 20

`forum_post_update()` (py-booru.api_danbooru.DanbooruApi_Mixin method), 20

`forum_topic_create()` (py-booru.api_danbooru.DanbooruApi_Mixin method), 19

`forum_topic_delete()` (py-booru.api_danbooru.DanbooruApi_Mixin method), 19

`forum_topic_list()` (py-booru.api_danbooru.DanbooruApi_Mixin method), 19

`forum_topic_show()` (py-booru.api_danbooru.DanbooruApi_Mixin method), 19

`forum_topic_undelete()` (py-booru.api_danbooru.DanbooruApi_Mixin method), 19

`forum_topic_update()` (py-booru.api_danbooru.DanbooruApi_Mixin method), 19

`hash_string` (`pybooru.moebooru.Moebooru` attribute), 21

H

L

`last_call` (`pybooru.danbooru.Danbooru` attribute), 5
`last_call` (`pybooru.moebooru.Moebooru` attribute), 21
`last_call` (`pybooru.pybooru._Pybooru` attribute), 29

M

`Moebooru` (class in `pybooru.moebooru`), 20
`MoebooruApi_Mixin` (class in `pybooru.api_moebooru`), 22

N

`note_create()` (`pybooru.api_danbooru.DanbooruApi_Mixin` method), 14
`note_create_update()` (`pybooru.api_moebooru.MoebooruApi_Mixin` method), 27
`note_delete()` (`pybooru.api_danbooru.DanbooruApi_Mixin` method), 14
`note_history()` (`pybooru.api_moebooru.MoebooruApi_Mixin` method), 27
`note_list()` (`pybooru.api_danbooru.DanbooruApi_Mixin` method), 14
`note_list()` (`pybooru.api_moebooru.MoebooruApi_Mixin` method), 26
`note_revert()` (`pybooru.api_danbooru.DanbooruApi_Mixin` method), 15
`note_revert()` (`pybooru.api_moebooru.MoebooruApi_Mixin` method), 27
`note_search()` (`pybooru.api_moebooru.MoebooruApi_Mixin` method), 26
`note_show()` (`pybooru.api_danbooru.DanbooruApi_Mixin` method), 14
`note_update()` (`pybooru.api_danbooru.DanbooruApi_Mixin` method), 14
`note_versions()` (`pybooru.api_danbooru.DanbooruApi_Mixin` method), 15

P

`password` (`pybooru.moebooru.Moebooru` attribute), 21
`pool_add_post()` (`pybooru.api_moebooru.MoebooruApi_Mixin` method), 28
`pool_create()` (`pybooru.api_danbooru.DanbooruApi_Mixin` method), 16

`pool_create()` (`pybooru.api_moebooru.MoebooruApi_Mixin` method), 28
`pool_delete()` (`pybooru.api_danbooru.DanbooruApi_Mixin` method), 16
`pool_destroy()` (`pybooru.api_moebooru.MoebooruApi_Mixin` method), 28
`pool_list()` (`pybooru.api_danbooru.DanbooruApi_Mixin` method), 15
`pool_list()` (`pybooru.api_moebooru.MoebooruApi_Mixin` method), 27
`pool_posts()` (`pybooru.api_moebooru.MoebooruApi_Mixin` method), 28
`pool_remove_post()` (`pybooru.api_moebooru.MoebooruApi_Mixin` method), 28
`pool_revert()` (`pybooru.api_danbooru.DanbooruApi_Mixin` method), 16
`pool_show()` (`pybooru.api_danbooru.DanbooruApi_Mixin` method), 16
`pool_undelete()` (`pybooru.api_danbooru.DanbooruApi_Mixin` method), 16
`pool_update()` (`pybooru.api_danbooru.DanbooruApi_Mixin` method), 16
`pool_update()` (`pybooru.api_moebooru.MoebooruApi_Mixin` method), 28
`pool_versions()` (`pybooru.api_danbooru.DanbooruApi_Mixin` method), 16
`post_appeals_create()` (`pybooru.api_danbooru.DanbooruApi_Mixin` method), 8
`post_appeals_list()` (`pybooru.api_danbooru.DanbooruApi_Mixin` method), 8
`post_appeals_show()` (`pybooru.api_danbooru.DanbooruApi_Mixin` method), 8
`post_copy_notes()` (`pybooru.api_danbooru.DanbooruApi_Mixin` method), 7
`post_create()` (`pybooru.api_moebooru.MoebooruApi_Mixin` method), 22
`post_destroy()` (`pybooru.api_moebooru.MoebooruApi_Mixin` method), 23
`post_flag_create()` (`pybooru.api_danbooru.DanbooruApi_Mixin` method), 16

<i>booru.api_danbooru.DanbooruApi_Mixin</i> <i>method</i>), 8	PybooruHTTPError, 30
post_flag_list() (py- <i>booru.api_danbooru.DanbooruApi_Mixin</i> <i>method</i>), 8	S
post_flag_show() (py- <i>booru.api_danbooru.DanbooruApi_Mixin</i> <i>method</i>), 8	site_name (<i>pybooru.danbooru.Danbooru</i> attribute), 5
post_list() (<i>pybooru.api_danbooru.DanbooruApi_Mixin</i> <i>method</i>), 6	site_name (<i>pybooru.moebooru.Moebooru</i> attribute), 20, 21
post_list() (<i>pybooru.api_moebooru.MoebooruApi_Mixin</i> <i>method</i>), 22	site_name (<i>pybooru.pybooru._Pybooru</i> attribute), 29
post_mark_translated() (py- <i>booru.api_danbooru.DanbooruApi_Mixin</i> <i>method</i>), 7	site_url (<i>pybooru.danbooru.Danbooru</i> attribute), 5
post_revert() (py- <i>booru.api_danbooru.DanbooruApi_Mixin</i> <i>method</i>), 7	site_url (<i>pybooru.moebooru.Moebooru</i> attribute), 21
post_revert_tags() (py- <i>booru.api_moebooru.MoebooruApi_Mixin</i> <i>method</i>), 23	site_url (<i>pybooru.pybooru._Pybooru</i> attribute), 29
post_show() (<i>pybooru.api_danbooru.DanbooruApi_Mixin</i> <i>method</i>), 6	T
post_unvote() (py- <i>booru.api_danbooru.DanbooruApi_Mixin</i> <i>method</i>), 7	tag_aliases() (py- <i>booru.api_danbooru.DanbooruApi_Mixin</i> <i>method</i>), 17
post_update() (py- <i>booru.api_danbooru.DanbooruApi_Mixin</i> <i>method</i>), 7	tag_implications() (py- <i>booru.api_danbooru.DanbooruApi_Mixin</i> <i>method</i>), 17
post_update() (py- <i>booru.api_moebooru.MoebooruApi_Mixin</i> <i>method</i>), 23	tag_list() (<i>pybooru.api_danbooru.DanbooruApi_Mixin</i> <i>method</i>), 16
post_versions_list() (py- <i>booru.api_danbooru.DanbooruApi_Mixin</i> <i>method</i>), 8	tag_list() (<i>pybooru.api_moebooru.MoebooruApi_Mixin</i> <i>method</i>), 24
post_versions_show() (py- <i>booru.api_danbooru.DanbooruApi_Mixin</i> <i>method</i>), 8	tag_related() (py- <i>booru.api_danbooru.DanbooruApi_Mixin</i> <i>method</i>), 17
post_versions_undo() (py- <i>booru.api_danbooru.DanbooruApi_Mixin</i> <i>method</i>), 9	tag_related() (py- <i>booru.api_moebooru.MoebooruApi_Mixin</i> <i>method</i>), 24
post_vote() (<i>pybooru.api_danbooru.DanbooruApi_Mixin</i> <i>method</i>), 7	tag_show() (<i>pybooru.api_danbooru.DanbooruApi_Mixin</i> <i>method</i>), 17
post_vote() (<i>pybooru.api_moebooru.MoebooruApi_Mixin</i> <i>method</i>), 23	tag_update() (<i>pybooru.api_danbooru.DanbooruApi_Mixin</i> <i>method</i>), 17
pybooru.api_danbooru (module), 6	tag_update() (<i>pybooru.api_moebooru.MoebooruApi_Mixin</i> <i>method</i>), 24
pybooru.api_moebooru (module), 22	U
pybooru.danbooru (module), 5	upload_create() (py- <i>booru.api_danbooru.DanbooruApi_Mixin</i> <i>method</i>), 9
pybooru.exceptions (module), 30	upload_list() (py- <i>booru.api_danbooru.DanbooruApi_Mixin</i> <i>method</i>), 9
pybooru.moebooru (module), 20	upload_show() (py- <i>booru.api_danbooru.DanbooruApi_Mixin</i> <i>method</i>), 9
pybooru.pybooru (module), 29	user_list() (<i>pybooru.api_danbooru.DanbooruApi_Mixin</i> <i>method</i>), 15
pybooru.resources (module), 31	user_search() (py- <i>booru.api_moebooru.MoebooruApi_Mixin</i> <i>method</i>), 27
PybooruAPIError, 31	user_show() (<i>pybooru.api_danbooru.DanbooruApi_Mixin</i> <i>method</i>), 15
PybooruError, 30	username (<i>pybooru.danbooru.Danbooru</i> attribute), 5
	username (<i>pybooru.moebooru.Moebooru</i> attribute), 21

username (*pybooru.pybooru._Pybooru* attribute), 29

W

`wiki_create()` (*pybooru.api_danbooru.DanbooruApi_Mixin* method), 18

`wiki_create()` (*pybooru.api_moebooru.MoebooruApi_Mixin* method), 26

`wiki_delete()` (*pybooru.api_danbooru.DanbooruApi_Mixin* method), 18

`wiki_destroy()` (*pybooru.api_moebooru.MoebooruApi_Mixin* method), 26

`wiki_history()` (*pybooru.api_moebooru.MoebooruApi_Mixin* method), 26

`wiki_list()` (*pybooru.api_danbooru.DanbooruApi_Mixin* method), 17

`wiki_list()` (*pybooru.api_moebooru.MoebooruApi_Mixin* method), 25

`wiki_lock()` (*pybooru.api_moebooru.MoebooruApi_Mixin* method), 26

`wiki_revert()` (*pybooru.api_danbooru.DanbooruApi_Mixin* method), 18

`wiki_revert()` (*pybooru.api_moebooru.MoebooruApi_Mixin* method), 26

`wiki_show()` (*pybooru.api_danbooru.DanbooruApi_Mixin* method), 18

`wiki_show()` (*pybooru.api_moebooru.MoebooruApi_Mixin* method), 26

`wiki_unlock()` (*pybooru.api_moebooru.MoebooruApi_Mixin* method), 26

`wiki_update()` (*pybooru.api_danbooru.DanbooruApi_Mixin* method), 18

`wiki_update()` (*pybooru.api_moebooru.MoebooruApi_Mixin* method), 26

`wiki_versions_list()` (*pybooru.api_danbooru.DanbooruApi_Mixin* method), 18

`wiki_versions_show()` (*pybooru.api_danbooru.DanbooruApi_Mixin* method), 19