# pixiv-api

*Release 1.0.0*

**Dec 05, 2022**

# Contents

A documented, idiomatic, and tested wrapper library around Pixiv's App API.

Supports Python 3.6+

Install with:

```
$ pip install pixiv-api
```

# CHAPTER 1

# Quickstart

To start making requests to the Pixiv API, instantiate a client object.

```
from pixivapi import Client

client = Client()
```

The client can be authenticated to Pixiv's API in multiple ways. One is by logging in with a username and password:

```
client.login('username', 'password')
```

And another is with a refresh token.

```
client.authenticate('refresh_token')
```

Once authenticated, a refresh token can be saved for future authorizations.

```
refresh_token = client.refresh_token
```

After authenticating, the client can begin making requests to all of the Pixiv endpoints. For example, the following code block downloads an image from Pixiv.

```
from pathlib import Path
from pixivapi import Size

illustration = client.fetch_illustration(75523989)
illustration.download(
    directory=Path.home() / 'my_pixiv_images',
    size=Size.ORIGINAL,
)
```

And the next code block downloads all illustrations of an artist.

```
from pathlib import Path
from pixivapi import Size
```

(continues on next page)

(continued from previous page)

```
artist_id = 2188232
directory = Path.home() / 'wlop'

response = client.fetch_user_illustrations(artist_id)
while True:
    for illust in response['illustrations']:
        illust.download(directory=directory, size=Size.ORIGINAL)

    if not response['next']:
        break

    response = client.fetch_user_illustrations(
        artist_id,
        offset=response['next'],
    )
```

CHAPTER 2

# API

## 2.1 Client

The client module contains a class that exposes a pythonic API to Pixiv's JSON API and encapsulates the translation logic between the two APIs.

**class** pixivapi.client.**Client**(*language='English'*, *client_id='KzEZED7aC0vird8jWyHM38mXjNTY'*, *client_secret='W9JZoJe00qPvJsiyCGT3CCtC6ZUtdpKpzMbNlUGP'*)

A client for the Pixiv API.

**Variables**

- **language** (*str*) – The language tag translations should be in.
- **client_id** (*str*) – The client ID. Typically, leaving this as default is ok.
- **client_secret** (*str*) – The client secret. Typically, leaving this as default is ok.
- **account** (Account) – Basic details of the logged in account.
- **access_token** (*str*) – The access token used to authorize requests.
- **refresh_token** (*str*) – The refresh token used to obtain new access tokens.
- **session** (*requests.Session*) – The requests session.

**download**(*url*, *destination*, *referer='https://pixiv.net'*)

Download a file to a given destination. This method uses the client's access token if available.

**Parameters**

- **url** (*str*) – The URL of the file.
- **destination** – The destination file. Must be writeable.
- **referer** (*str*) – The Referer header.

**Raises**

- **FileNotFoundError** – If the destination's directory does not exist.

- **PermissionError** – If the destination cannot be written to.

**login** (*username*, *password*)
Log in with username and password to fetch an access token and a refresh token. Assigns the tokens to instance variables.

**Parameters**

- **username** (*str*) – Your username.
- **password** (*str*) – Your password.

**Raises** ***LoginError*** – If login fails.

**authenticate** (*refresh_token*)
Use a refresh token to obtain a new access token. Assigns both tokens to instance variables.

**Parameters** **refresh_token** (*str*) – The refresh token.

**Raises** ***LoginError*** – If authentication fails.

**search_illustrations** (*word*, *search_target=<SearchTarget.TAGS_PARTIAL: 'partial_match_for_tags'>*, *sort=<Sort.DATE_DESC: 'date_desc'>*, *duration=None*, *offset=None*)
Search the illustrations. A maximum of 30 illustrations are returned in one response.

**Parameters**

- **word** (*str*) – The search term.
- **search_target** (SearchTarget) – The target for the search term.
- **sort** (Sort) – How to sort the illustrations.
- **duration** (Duration) – An optional max-age for the illustrations.
- **offset** (*int*) – The number of illustrations to offset by.

**Returns** A dictionary containing the searched illustrations, the offset for the next page of search images (None if there is no next page), and the search span limit.

```
{
    'illustrations': [Illustration, ...],  # List of illustrations.
    'next': 30,  # Offset to get the next page of illustrations.
    'search_span_limit': 31536000,
}
```

**Return type** dict

**Raises**

- **requests.RequestException** – If the request fails.
- ***BadApiResponse*** – If the response is not valid JSON.

**fetch_illustration** (*illustration_id*)
Fetch the details of a single illustration.

**Parameters** **illustration_id** (*int*) – The ID of the illustration.

**Returns** An illustration object.

**Return type** *Illustration*

**Raises**

- **requests.RequestException** – If the request fails.
- ***BadApiResponse*** – If the response is not valid JSON.

**fetch_illustration_comments**(*illustration_id*, *offset=None*, *include_total_comments=False*)

Fetch the comments of an illustration. A maximum of 30 comments are returned in one response.

**Note:** The `total_comments` key does not equal the number of comments that will be returned by the API. If requesting all the comments, use the `next` key to determine whether or not to continue, not the `total_comments` key.

**Parameters**

- **illustration_id** (*int*) – ID of the illustration.
- **offset** (*int*) – Number of comments to offset by.
- **include_total_comments** (*bool*) – Whether or not to include a the total number of comments on the illustration. If set to False, the `total_comments` key in the response will be `0`.

**Returns** A dictionary containing the comments, the offset for the next page of comments, and the total number of comments.

```
{
    'comments': [Comment, ...],  # List of comments.
    'next': 30,  # Offset to get the next page of comments.
    'total_comments': 142,
}
```

**Return type** dict

**Raises**

- **requests.RequestException** – If the request fails.
- ***BadApiResponse*** – If the response is not valid JSON.

**fetch_illustration_related**(*illustration_id*, *offset=None*)

Fetch illustrations related to a specified illustration. A maximum of 30 illustrations are returned in one response.

**Parameters**

- **illustration_id** (*int*) – ID of the illustration.
- **offset** (*int*) – Illustrations to offset by.

**Returns** A dictionary containing the related illustrations and the offset for

the next page of illustrations.

```
{
    'illustrations': [Illustration, ...],  # List of illustrations.
    'next': 30,  # Offset to get the next page of illustrations.
}
```

**Return type** dict

**Raises**

- **requests.RequestException** – If the request fails.
- ***BadApiResponse*** – If the response is not valid JSON.

**fetch_illustrations_following**(*visibility=<Visibility.PUBLIC: 'public'>*, *offset=None*)

Fetch new illustrations from followed artists. A maximum of 30 illustrations are returned in one response.

**Parameters**

- **visibility** (Visibility) – Visibility of the followed artist; PUBLIC if publicly followed; PRIVATE if privately.
- **offset** (*int*) – The number of illustrations to offset by.

**Returns** A dictionary containing the new illustrations and the offset for the next page of illustrations.

```
{
    'illustrations': [Illustration, ...],  # List of illustrations.
    'next': 30,  # Offset to get the next page of illustrations.
}
```

**Return type** dict

**Raises**

- **requests.RequestException** – If the request fails.
- ***BadApiResponse*** – If the response is not valid JSON.

**fetch_illustrations_recommended**(*content_type=<ContentType.ILLUSTRATION: 'illust'>*, *include_ranking_illustrations=False*, *max_bookmark_id_for_recommend=None*, *min_bookmark_id_for_recent_illustrations=None*, *offset=None*, *bookmark_illust_ids=None*, *include_ranking_label=True*)

Fetch one's recommended illustrations.

**Parameters**

- **content_type** (ContentType) – The type of content to fetch. Accepts ILLUSTRATION and MANGA.
- **include_ranking_illustrations** (*bool*) – If True, the top 10 ranked illustrations daily are included in the response. If False, the ranking_illustrations key in the response dict will be empty.
- **max_bookmark_id_for_recommend** (*int*) – The maximum bookmark ID for recommended illustrations, used for changing the returned illustrations.
- **min_bookmark_id_for_recent_illustrations** (*int*) – The minimum bookmark ID for recent illustrations, used for changing the returned illustrations.
- **offset** (*int*) – The number of illustrations to offset by.
- **bookmark_illust_ids** (*list*) – A list of illustration IDs.
- **include_ranking_label** (*bool*) – Whether or not to include the ranking label.

**Returns** A dictionary containing the recommended illustrations and the parameters for the next page of illustrations.

```
{
    'contest_exists': False,  # Does a contest exist?
    'illustrations': [Illustration, ...],  # List of illustrations.
    'next': {  # Parameters to get the next set of illustrations.
```

(continues on next page)

(continued from previous page)

```
        'min_bookmark_id_for_recent_illustrations': 6277740037,
        'max_bookmark_id_for_recommend': 6268205545,
        'offset': 0,
    },
    'ranking_illustrations': [Illustration, ...] # Ranking illust.
}
```

**Return type** dict

**Raises**

- **requests.RequestException** – If the request fails.
- ***BadApiResponse*** – If the response is not valid JSON.

**fetch_illustrations_ranking**(*mode=<RankingMode.DAY: 'day'>*, *date=None*, *offset=None*)

Fetch the ranking illustrations. A maximum of 30 illusrations are returned in one response.

**Parameters**

- **mode** (RankingMode) – The ranking list to fetch.
- **date** (*str*) – The date of the list, in `%Y-%m-%d` format.
- **offset** (*int*) – The number of illustrations to offset by.

**Returns** A dictionary containing the ranking illustrations and the offset for the next page of illustrations.

```
{
    'illustrations': [Illustration, ...],  # List of illustrations.
    'next': 30,  # Offset to get the next page of illustrations.
}
```

**Return type** dict

**Raises**

- **requests.RequestException** – If the request fails.
- ***BadApiResponse*** – If the response is not valid JSON.

**fetch_trending_tags**()

Fetch trending illustrations and tags.

**Returns** A list of dicts containing an illustration, the tag name, and the tag translation.

```
[
    {
        'illustration': Illustration,
        'tag': '',
        'translated_name': 'Kancolle',
    },
    ...
]
```

**Return type** dict

**Raises**

- **requests.RequestException** – If the request fails.
- ***BadApiResponse*** – If the response is not valid JSON.

**fetch_bookmark**(*illustration_id*)
Fetch details about a bookmarked illustration.

**Parameters** **illustration_id** (*int*) – The ID of the bookmarked illustration.

**Returns** A dictionary containing whether or not the illustration is bookmarked, the visibility of the bookmark, and a list of tags.

```
{
    'is_bookmarked': True,
    'visibility': Visibility.PUBLIC,
    'tags': [
        {
            'is_registered': False,
            'name': 'ghostblade',
        },
        ...
    ],
}
```

**Return type** dict

**Raises**

- **requests.RequestException** – If the request fails.
- ***BadApiResponse*** – If the response is not valid JSON.

**add_bookmark**(*illustration_id*, *visibility=<Visibility.PUBLIC: 'public'>*, *tags=None*)
Bookmark an illustration.

**Parameters**

- **illustration_id** (*int*) – The ID of the illustration.
- **visibility** (*Visibility*) – The visibility of the bookmark.
- **tags** (*list*) – The bookmark tags of the illustration.

**Raises** **requests.RequestException** – If the request fails.

**delete_bookmark**(*illustration_id*)
Delete a bookmark.

**Parameters** **illustration_id** (*int*) – The ID of the illustration.

**Raises** **requests.RequestException** – If the request fails.

**fetch_user**(*user_id*)
Fetch details about a Pixiv user.

**Parameters** **user_id** (*int*) – The ID of the user.

**Returns** A FullUser object.

**Return type** *FullUser*

**Raises**

- **requests.RequestException** – If the request fails.

- **BadApiResponse** – If the response is not valid JSON.

**fetch_user_illustrations**(*user_id*, *content_type=<ContentType.ILLUSTRATION: 'illust'>*, *offset=None*)

Fetch the illustrations posted by a user.

**Parameters**

- **user_id** (*int*) – The ID of the user.
- **content_type** (ContentType) – The type of content to fetch. Accepts ILLUSTRATION and MANGA.
- **offset** (*int*) – The number of illustrations/manga to offset by.

**Returns** A dictionary containing the user's illustrations and the offset to get the next page of their illustrations. If there is no next page, offset will be None.

```
{
    'illustrations': [Illustration, ...],  # List of illustrations.
    'next': 30,  # Offset to get the next page of illustrations.
}
```

**Return type** dict

**Raises**

- **requests.RequestException** – If the request fails.
- **BadApiResponse** – If the response is not valid JSON.

**fetch_user_bookmarks**(*user_id*, *visibility=<Visibility.PUBLIC: 'public'>*, *max_bookmark_id=None*, *tag=None*)

Fetch the illustrations bookmarked by a user. A maximum of 30 illustrations are returned in a response.

**Parameters**

- **user_id** (*int*) – The ID of the user.
- **visibility** (Visibility) – The visibility of the bookmarks. Applies only to requests for one's own bookmarks. If set to Visibility.PRIVATE for another user, their public bookmarks will be returned.
- **max_bookmark_id** (*int*) – The ID of the maximum bookmark, similar to offset for other endpoints.
- **tag** (*str*) – The bookmark tag to filter bookmarks by. These tags can be fetched from Client.fetch_user_bookmark_tags.

**Returns** A dictionary containing the user's bookmarks and the max_bookmark_id needed to get the next page of their bookmarks. If there is no next page, max_bookmark_id will be None.

```
{
    'illustrations': [Illustration, ...],  # List of illustrations.
    'next': 30,  # `max_bookmark_id` for the next page of bookmarks.
}
```

**Return type** dict

**Raises**

- **requests.RequestException** – If the request fails.

- ***BadApiResponse*** – If the response is not valid JSON.

**fetch_user_bookmark_tags**(*user_id*, *visibility=<Visibility.PUBLIC: 'public'>*, *offset=None*)
Fetch the bookmark tags that belong to the user. A maximum of 30 tags are returned in a response.

**Parameters**

- **user_id** (*int*) – The ID of the user whose bookmark tags to fetch.
- **visibility** (Visibility) – The visibility of the tags. Will raise an error if another user's private tags are requested.
- **offset** (*int*) – The number of tags to offset by.

**Returns** A dictionary containing the user's bookmark tags and the offset needed to get the next page of their bookmark tags. If there is no next page, `offset` will be `None`.

```
{
    'bookmark_tags': [  # List of bookmark tags.
        {
            'count': 5,  # Number of bookmarks with the tag.
            'name': 'a-bookmark-tag',
        },
        ...
    ],
    'next': 30,  # Offset for the next page of bookmark tags.
}
```

**Return type** dict

**Raises**

- **requests.RequestException** – If the request fails.
- ***BadApiResponse*** – If the response is not valid JSON or if another user's private tags are requested.

**fetch_following**(*user_id*, *visibility=<Visibility.PUBLIC: 'public'>*, *offset=None*)
Fetch the users that a user is following. A maximum of 30 users are returned in a response.

**Parameters**

- **user_id** (*int*) – The ID of the user.
- **visibility** (Visibility) – The visibility of the followed users. Applies only to one's own follows. If `Visibility.PRIVATE` is applied to another user, their publicly followed users will be returned.
- **offset** (*int*) – The number of users to offset by.

**Returns** A dictionary containing the a list of previews for the followed users and and the offset needed to get the next page of user previews. If there is no next page, `offset` will be `None`.

```
{
    'user_previews': [  # List of bookmark tags.
        {
            'illustrations': [  # Their 3 most recent illustrations.
                Illustration,
                Illustration,
                Illustration,
            ],
```

(continues on next page)

(continued from previous page)

```
            'is_muted': False,  # Are they muted?
            'novels': [   # Their 3 most recent novels.
                Novel,
                Novel,
                Novel,
            ],
            'user': User,  # Basic information about the user.
        },
        ...
    ],
    'next': 30,  # Offset for the next page of user previews.
}
```

**Return type** dict

**Raises**

- **requests.RequestException** – If the request fails.
- ***BadApiResponse*** – If the response is not valid JSON.

**fetch_followers**(*offset=None*)

Fetch the users that are following the requesting user.

**Parameters** **offset** (*int*) – The number of users to offset by.

**Returns** A dictionary containing the a list of previews for the users that follow the the requesting user and and the offset needed to get the next page of user previews. If there is no next page, `offset` will be `None`.

```
{
    'user_previews': [  # List of bookmark tags.
        {
            'illustrations': [  # Their 3 most recent illustrations.
                Illustration,
                Illustration,
                Illustration,
            ],
            'is_muted': False,  # Are they muted?
            'novels': [   # Preview of their novels.
                Novel,
                Novel,
                Novel,
            ],
            'user': User,  # Basic information about the user.
        },
        ...
    ],
    'next': 30,  # Offset for the next page of user previews.
}
```

**Return type** dict

**Raises**

- **requests.RequestException** – If the request fails.
- ***BadApiResponse*** – If the response is not valid JSON.

## 2.2 Models

The models module contains dataclasses that wrap entities returned from Pixiv's API.

**class** pixivapi.models.**User**(*account*, *id*, *name*, *profile_image_urls*, *is_followed=None*)
Bases: object

A model that represents a user. Not all instance variables will be populated; the variables that are populated depends on the endpoint that the user is fetched from.

Typically, the profile_image_urls dict will contain a 'medium' key.

**Variables**

- **account** (*str*) – Their account name
- **id** (*int*) – Their account ID
- **name** (*str*) – Their display name
- **profile_image_urls** (*dict*) – A dictionary of URLs for their profile image. The keys are the size and the value is the URL.
- **is_followed** (*bool*) – If the user is followed. If the endpoint does not return this (e.g. comments), it will be None.

**class** pixivapi.models.**Account**(*profile_image_urls*, *account*, *id*, *name*, *mail_address*, *is_premium*, *x_restrict*, *is_mail_authorized*)
Bases: *pixivapi.models.User*

**A model for the authenticating user that inherits from User. The properties** present in the User model are also present here.

The profile images have the sizes 16x16, 50x50, 170x170.

**Variables**

- **mail_address** (*str*) – The user's email
- **is_premium** (*bool*) – Whether or not user has Pixiv premium
- **x_restrict** (*int*) – User's x restriction
- **is_mail_authorized** (*bool*) – Whether user's email was authorized

**class** pixivapi.models.**FullUser**(*account*, *id*, *name*, *profile_image_urls*, *is_followed=None*, *comment=None*, *profile=None*, *profile_publicity=None*, *workspace=None*)
Bases: *pixivapi.models.User*

This model inherits the properties that the User model has.

**Variables**

- **comment** (*str*) – The comment on the user's account. Only provided when fetching user via Client.fetch_user.
- **profile** (*dict*) – Profile information fetched from the Client.fetch_user endpoint. Example below.

```
{
    'address_id': 01,
    'background_image_url': None,
    'birth': '',
```

(continues on next page)

(continued from previous page)

```
    'birth_day': '01-01',
    'birth_year': 0,
    'country_code': 'CN',
    'gender': '',
    'is_premium': True,
    'is_using_custom_profile_image': True,
    'job': 'Something',
    'job_id': 1,
    'pawoo_url': (
        'https://pawoo.net/oauth_authentications/123?provider=pixiv
↪'
    ),
    'region': 'China',
    'total_follow_users': 0,
    'total_illust_bookmarks_public': 1,
    'total_illust_series': 0,
    'total_illusts': 0,
    'total_manga': 0,
    'total_mypixiv_users': 0,
    'total_novel_series': 0,
    'total_novels': 0,
    'twitter_account': 'twittername',
    'twitter_url': 'https://twitter.com/twittername',
    'webpage': 'https://webpage.com'
}
```

- **profile_publicity** (*dict*) – A dictionary detailling which parts of the user's profile are public and which are private. Only provided from the `Client.fetch_user` endpoint. Example below.

```
{
    'birth_day': 'public',
    'birth_year': 'public',
    'gender': 'public',
    'job': 'public',
    'pawoo': True,
    'region': 'public'
}
```

- **workspace** (*dict*) – A dictionary containing information about the user's workspace. Only provided from the `Client.fetch_user` endpoint. Example below.

```
{
    'chair': '',
    'comment': '',
    'desk': '',
    'desktop': '',
    'monitor': '',
    'mouse': '',
    'music': '',
    'pc': '',
    'printer': '',
    'scanner': '',
    'tablet': '',
    'tool': '',
    'workspace_image_url': None
}
```

**class** pixivapi.models.**Illustration**(*caption*, *create_date*, *height*, *id*, *image_urls*, *is_bookmarked*, *is_muted*, *meta_pages*, *meta_single_page*, *page_count*, *restrict*, *sanity_level*, *series*, *tags*, *title*, *tools*, *total_bookmarks*, *total_view*, *type*, *user*, *visible*, *width*, *x_restrict*, *client=None*, *total_comments=None*, *illust_ai_type=None*, *illust_book_style=None*, *comment_access_control=None*)

Bases: object

The illustration models encapsulates an illustration and provides methods for convenient fetching of related objects.

**Variables**

- **caption** (*str*) – Caption
- **create_date** (*datetime.datetime*) – Creation date
- **height** (*int*) – Height
- **id** (*int*) – ID
- **image_urls** (*dict*) – A dict of Image URLs mapping the Size enum to the URL. If the image has multiple pages, *Size.ORIGINAL* will be None.
- **is_bookmarked** (*bool*) – If the image is bookmarked.
- **is_muted** (*bool*) – If the image is muted.
- **meta_pages** (*list*) – If the image has multiple images, list this will be a list of dicts mapping the Size enum to image urls. If not, this will be an empty list.
- **page_count** (*int*) – The number of pages.
- **restrict** (*int*) – The restriction.
- **sanity_level** (*int*) – The sanity level.
- **series** – If the illustration is in a series, this will be a dict with the `id` and `title` key/value pairs of the series. If the illustration is not in a series, this will be `None`.
- **tags** (*list*) – A list of dicts containing two keys: `name` and `translated_name`. The `translated_name` will be `None` if the client language is not set.
- **title** (*str*) – The title of the work.
- **tools** (*list*) – The listed tools used to create the illustration.
- **total_bookmarks** (*int*) – The number of times the illustration has been bookmarked.
- **total_view** (*int*) – The number of times the illustration has been viewed.
- **type** (ContentType) – The content type (illustration, manga).
- **user** (User) – The artist of the image.
- **visible** (*bool*) – The visibility.
- **width** (*int*) – The width of the illustration.
- **x_restrict** (*int*) – The x restrict.
- ***client*** (Client) – The client used to fetch the image information.
- **total_comments** (*int*) – The total number of comments on the illustration. This value may be `None` depending on which method this illustration was fetched from. For example, this value is not returned when searching illustrations.

**download**(*directory*, *size=<Size.ORIGINAL: 'original'>*, *filename=None*)
Download the illustration to the desired directory. If the illustration has multiple pages, a folder will be created and the images placed inside.

**Parameters**

- **directory** (*pathlib.Path*) – The illustration will be downloaded to this directory.
- **size** (Size) – The size of the image to download.
- **filename** – Do not include the file extension. This will be the filename of a single-page illustration and the folder name of a multi-page illustration. By default this will be the ID of the illustration.

**Raises**

- **requests.RequestException** – If the request fails.
- **FileNotFoundError** – If the destination's directory does not exist.
- **PermissionError** – If the destination cannot be written to.

**class** pixivapi.models.**Novel**(*caption*, *create_date*, *id*, *image_urls*, *is_bookmarked*, *is_muted*, *is_mypixiv_only*, *is_x_restricted*, *is_original*, *page_count*, *restrict*, *series*, *tags*, *text_length*, *title*, *total_bookmarks*, *total_comments*, *total_view*, *user*, *visible*, *x_restrict*, *client=None*)

Bases: object

A model that encapsulates a novel.

**Variables**

- **caption** (*str*) – Caption
- **create_date** (*datetime.datetime*) – Creation date
- **id** (*int*) – ID
- **image_urls** (*dict*) – A dict of Image URLs mapping the Size enum to the URL. There is no Size.ORIGINAL.
- **is_bookmarked** (*bool*) – If the novel is bookmarked.
- **is_muted** (*bool*) – If the novel is muted.
- **is_mypixiv_only** (*bool*) – If the novel is mypixiv only.
- **is_x_restricted** (*bool*) – If the novel is X restricted.
- **is_original** (*bool*) – If the novel is an original.
- **page_count** (*int*) – The number of pages.
- **restrict** (*int*) – The restriction.
- **series** – If the novel is in a series, this will be a dict with the id and title key/value pairs of the series. If the novel is not in a series, this will be None.
- **tags** (*list*) – A list of dicts containing three keys: name, translated_name, and added_by_uploaded_user. The translated_name will be None if the client language is not set.
- **text_length** (*int*) – The length of the novel.
- **title** (*str*) – The title of the novel.
- **total_bookmarks** (*int*) – The number of times the novel has been bookmarked.

- **total_comments** (*int*) – The total number of comments on the novel.
- **total_view** (*int*) – The number of times the novel has been viewed.
- **user** (User) – The author of the novel.
- **visible** (*bool*) – The visibility.
- **x_restrict** (*int*) – The X restrict.
- ***client*** (Client) – The client used to fetch the novel information.

**class** pixivapi.models.**Comment**(*comment*, *date*, *id*, *parent_comment*, *user*, *client=None*)
Bases: object

A model that encapsulates a comment.

**Variables**

- **comment** (*str*) – Content of the comment
- **date** (*datetime.datetime*) – Date the comment was posted
- **id** (*int*) – ID of the comment
- **parent_comment** (Comment) – A parent comment to this comment (can be None)
- **user** (User) – The poster of the comment. Does not return whether or not the user is followed.

## 2.3 Enums

The enums module defines the constants used in this library's API.

**class** pixivapi.enums.**ContentType**
This Enum represents the various types of content that are present on Pixiv.

**ILLUSTRATION = 'illust'**

**MANGA = 'manga'**

**UGOIRA = 'ugoira'**

**NOVEL = 'novel'**

**class** pixivapi.enums.**Duration**
This Enum is used when searching Pixiv to limit the age of the returned results.

**LAST_DAY = 'within_last_day'**

**LAST_WEEK = 'within_last_week'**

**LAST_MONTH = 'within_last_month'**

**class** pixivapi.enums.**RankingMode**
This Enum is used to specify which ranking list of illustrations should be fetched.

**DAY = 'day'**

**WEEK = 'week'**

**MONTH = 'month'**

**DAY_MALE = 'day_male'**

**DAY_FEMALE = 'day_female'**

**WEEK_ORIGINAL = 'week_original'**

**WEEK_ROOKIE = 'week_rookie'**

**DAY_MANGA = 'day_manga'**

**class** pixivapi.enums.**SearchTarget**
This Enum determines how the search should match the searched words to the possible results.

**TAGS_PARTIAL = 'partial_match_for_tags'**

**TAGS_EXACT = 'exact_match_for_tags'**

**TITLE_AND_CAPTION = 'title_and_caption'**

**class** pixivapi.enums.**Size**
This Enum represents the possible sizes of an image. `ORIGINAL` has the best quality.

**LARGE = 'large'**

**MEDIUM = 'medium'**

**ORIGINAL = 'original'**

**SQUARE_MEDIUM = 'square_medium'**

**class** pixivapi.enums.**Sort**
This Enum determines how the search results are sorted by date; either oldest first or newest first.

**DATE_DESC = 'date_desc'**

**DATE_ASC = 'date_asc'**

**class** pixivapi.enums.**Visibility**
This Enum represents the visibility restrictions that a Pixiv user can enforce on their bookmarks, followed users, etc.

**PUBLIC = 'public'**

**PRIVATE = 'private'**

## 2.4 Exceptions

The errors module defines the errors used in this library's API.

**exception** pixivapi.errors.**PixivError**
Bases: `Exception`

**exception** pixivapi.errors.**LoginError**
Bases: *pixivapi.errors.PixivError*

**exception** pixivapi.errors.**AuthenticationRequired**
Bases: *pixivapi.errors.PixivError*

**exception** pixivapi.errors.**BadApiResponse**
Bases: *pixivapi.errors.PixivError*

CHAPTER 3

# Changelog

## 3.1 v1.0.0

- Fixed Py3.6 bug where datetimes failed to deserialize.
- Add reprs to model classes.
- Add a test suite and some CI checks.

## 3.2 v0.3.7

- Add ability to specify tags when adding a bookmark.

## 3.3 v0.3.6

- Fix inability to login.

## 3.4 v0.3.5

- Fix issue with offset not working in *fetch_illustrations_following*.

## 3.5 v0.3.4

- Fix issue with Python 3.6 compatibility wrt. datetime module.

## 3.6 v0.3.3

- Fix arguments of Novel class instantiation.

## 3.7 v0.3.1

- Fix quickstart example documentation.

## 3.8 v0.3.0

- Update authentication in response to Pixiv's changes.

## 3.9 v0.2.0

- Change `Client.account` from a dict to an `Account` model.
- Remove `None` attributes from User that only applied to responses from `Client.fetch_user` and move them to a `FullUser` subclass.
- Change return type of `Client.fetch_user` to a `FullUser`. No attributes were changed.

# Python Module Index

## p

# Index

## N



## O



## P



## R



## S



## T



## U



## V



## W