myanimelistpy

Release 0.1

Allan Capistrano

Jun 28, 2023

CONTENTS

A library to use the MyAnimeList API more easily.

CHAPTER

ONE

GETTING STARTED

1.1 Client ID

First, you need to get a **Client ID** in the MyAnimeList platform. To do this, follow these steps:

1. If you haven't already, create an account on the MyAnimeList platform;
2. After that, go to **Account Settings** page;
3. In the top menu, click in the **API** option;
4. It's important that you read the information on the API page, after that, click on **Create ID** button;
5. Fill in the required fields and agree to the **API License and Developer Agreement** after reading it;
6. After creating your **Clients Accessing the MAL API**, click on **Edit** button, then copy and save your **Client ID**.

1.2 Installation

Click here to check how to install the myanimelistpy library.

1.3 Examples

1.3.1 Anime List

The sections below will show you how to get the anime list using different methods.

`getAnimeList()`

Basic example

```
from myanimelistpy.myanimelist import MyAnimeList

CLIENT_ID = "YOUR_MY_ANIME_LIST_CLIENT_ID"

if __name__ == "__main__":
    my_anime_list = MyAnimeList(client_id=CLIENT_ID)

```

(continues on next page)

(continued from previous page)

```
    anime_list = my_anime_list.getAnimeList(
        anime_name = "Hunter x Hunter",
        limit      = 2
    )

    for anime in anime_list:
        print(f"Id: {anime.getId()}")
        print(f"Title: {anime.getTitle()}")
        print(f"Main Picture (medium): {anime.getMainPicture().getMedium()}\n")
```

Output

```
Id: 11061
Title: Hunter x Hunter (2011)
Main Picture (medium): https://api-cdn.myanimelist.net/images/anime/1337/99013.jpg

Id: 136
Title: Hunter x Hunter
Main Picture (medium): https://api-cdn.myanimelist.net/images/anime/8/19473.jpg
```

Request with fields

```
from myanimelistpy.myanimelist import MyAnimeList

CLIENT_ID = "YOUR_MY_ANIME_LIST_CLIENT_ID"

if __name__ == "__main__":
    my_anime_list = MyAnimeList(client_id=CLIENT_ID)

    anime_list = my_anime_list.getAnimeList(
        anime_name = "Hunter x Hunter",
        limit      = 2,
        fields     = ["rank", "status"]
    )

    for anime in anime_list:
        print(f"Id: {anime.getId()}")
        print(f"Title: {anime.getTitle()}")
        print(f"Main Picture (medium): {anime.getMainPicture().getMedium()}")
        print(f"Rank: {anime.getRank()}")
        print(f"Status: {anime.getStatus()}\n")
```

Output

```
Id: 11061
Title: Hunter x Hunter (2011)
Main Picture (medium): https://api-cdn.myanimelist.net/images/anime/1337/99013.jpg
Rank: 9
```

(continues on next page)

(continued from previous page)

```
Status: Finished airing

Id: 136
Title: Hunter x Hunter
Main Picture (medium): https://api-cdn.myanimelist.net/images/anime/8/19473.jpg
Rank: 165
Status: Finished airing
```

Tip: We strongly recommend using a `.env` file or something similar to store your **Client ID**. You can use the python-dotenv library to do this. Click here to check out their documentation.

```python
from os import getenv
from dotenv import load_dotenv

from myanimelistpy.myanimelist import MyAnimeList

load_dotenv()

CLIENT_ID = getenv("CLIENT_ID")

if __name__ == "__main__":
    my_anime_list = MyAnimeList(client_id=CLIENT_ID)

    anime_list = my_anime_list.getAnimeList(
        anime_name = "Hunter x Hunter",
        limit      = 2,
        fields     = ["rank", "status"]
    )

    for anime in anime_list:
        print(f"Id: {anime.getId()}")
        print(f"Title: {anime.getTitle()}")
        print(f"Main Picture (medium): {anime.getMainPicture().getMedium()}")
        print(f"Rank: {anime.getRank()}")
        print(f"Status: {anime.getStatus()}\n")
```

All the next examples will using the python-dotenv library, but feel free not to use it if you don't want to.

getAnimeListInDict()

```python
from os import getenv
from dotenv import load_dotenv

from myanimelistpy.myanimelist import MyAnimeList

load_dotenv()

CLIENT_ID = getenv("CLIENT_ID")
```

(continues on next page)

(continued from previous page)

```
9
10  if __name__ == "__main__":
11      my_anime_list = MyAnimeList(client_id=CLIENT_ID)
12
13      anime_list = my_anime_list.getAnimeListInDict(
14          anime_name = "Hunter x Hunter",
15          limit      = 2,
16          fields     = ["genres"]
17      )
18
19      for anime in anime_list:
20          print(str(anime) + "\n")
```

Output

```
{'node': {'id': 11061, 'title': 'Hunter x Hunter (2011)', 'main_picture': {'medium':
↪'https://api-cdn.myanimelist.net/images/anime/1337/99013.jpg', 'large': 'https://api-
↪cdn.myanimelist.net/images/anime/1337/99013l.jpg'}, 'genres': [{'id': 1, 'name':
↪'Action'}, {'id': 2, 'name': 'Adventure'}, {'id': 10, 'name': 'Fantasy'}, {'id': 27,
↪'name': 'Shounen'}]}}

{'node': {'id': 136, 'title': 'Hunter x Hunter', 'main_picture': {'medium': 'https://api-
↪cdn.myanimelist.net/images/anime/8/19473.jpg', 'large': 'https://api-cdn.myanimelist.
↪net/images/anime/8/19473l.jpg'}, 'genres': [{'id': 1, 'name': 'Action'}, {'id': 2,
↪'name': 'Adventure'}, {'id': 10, 'name': 'Fantasy'}, {'id': 27, 'name': 'Shounen'}]}}
```

Note: You can access dictionary properties using square brackets:

```
19  for anime in anime_list:
20      print(f"Title: {anime['node']['title']}")
21      print(f"Genres: {anime['node']['genres']}\n")
```

Output

```
Title: Hunter x Hunter (2011)
Genres: [{'id': 1, 'name': 'Action'}, {'id': 2, 'name': 'Adventure'}, {'id': 10, 'name':
↪'Fantasy'}, {'id': 27, 'name': 'Shounen'}]

Title: Hunter x Hunter
Genres: [{'id': 1, 'name': 'Action'}, {'id': 2, 'name': 'Adventure'}, {'id': 10, 'name':
↪'Fantasy'}, {'id': 27, 'name': 'Shounen'}]
```

`getAnimeListInJSON()`

```python
from os import getenv
from dotenv import load_dotenv

from myanimelistpy.myanimelist import MyAnimeList

load_dotenv()

CLIENT_ID = getenv("CLIENT_ID")

if __name__ == "__main__":
    my_anime_list = MyAnimeList(client_id=CLIENT_ID)

    anime_list = my_anime_list.getAnimeListInJSON(
        anime_name = "Hunter x Hunter",
        limit      = 2,
        fields     = ["num_episodes"]
    )

    print(anime_list)
```

Output


```
{"data":[{"node": {"id": 11061, "title": "Hunter x Hunter (2011)", "main_picture": {
↪"medium": "https://api-cdn.myanimelist.net/images/anime/1337/99013.jpg", "large":
↪"https://api-cdn.myanimelist.net/images/anime/1337/99013l.jpg"}, "num_episodes": 148}},
↪ {"node": {"id": 136, "title": "Hunter x Hunter", "main_picture": {"medium": "https://
↪api-cdn.myanimelist.net/images/anime/8/19473.jpg", "large": "https://api-cdn.
↪myanimelist.net/images/anime/8/19473l.jpg"}, "num_episodes": 62}}]}
```


Note: JSON Viewer


```
{
    "data": [
        {
            "node": {
                "id": 11061,
                "title": "Hunter x Hunter (2011)",
                "main_picture": {
                "medium": "https://api-cdn.myanimelist.net/images/anime/1337/99013.jpg",
                "large": "https://api-cdn.myanimelist.net/images/anime/1337/99013l.jpg"
                },
                "num_episodes": 148
            }
        },
        {
            "node": {
                "id": 136,
                "title": "Hunter x Hunter",
```


(continues on next page)

(continued from previous page)

```
                "main_picture": {
                "medium": "https://api-cdn.myanimelist.net/images/anime/8/19473.jpg",
                "large": "https://api-cdn.myanimelist.net/images/anime/8/19473l.jpg"
                },
                "num_episodes": 62
            }
        }
    ]
}
```

CHAPTER

TWO

REFERENCE

The following section outlines the reference of myanimelistpy.

2.1 MyAnimeList

class myanimelist.**MyAnimeList**(*client_id*)

Methods
getAnimeList
getAnimeListInDict
getAnimeListInJSON

Parameters:

- client_id (str) - MyAnimeList Client ID.

2.1.1 Methods

getAnimeList(*anime_name*, *limit=100*, *offset=0*, *fields=[]*)

Returns a list of anime by name.

Parameters:

- anime_name(str) - Name of the anime/series.
- limit(Optional[int]) - The maximum size of the list. The default value is 100.
- offset(Optional[int]) - The list offset. The default value is 0.
- fields(Optional[List [str]]) - List of fields used to show more information about the anime. If is empty, the default fields are id, title and main_picture.

Raises:

- ValueError - The 'field' is not a valid field!

Returns:
List of animes.

Return type:
List [*Anime*]

getAnimeListInDict(*anime_name*, *limit=100*, *offset=0*, *fields=[]*)

Returns a list of dictionaries containing the anime by name.

Parameters:

- anime_name(str) - Name of the anime/series.
- limit(Optional[int]) - The maximum size of the list. The default value is 100.
- offset(Optional[int]) - The list offset. The default value is 0.
- fields(Optional[List [str]]) - List of fields used to show more information about the anime. If is empty, the default fields are id, title and main_picture.

Raises:

- ValueError - The 'field' is not a valid field!

Returns:
List of animes in dict type.

Return type:
dict

getAnimeListInJSON(*anime_name*, *limit=100*, *offset=0*, *fields=[]*)

Returns a JSON stringified containing the list of anime by name.

Parameters:

- anime_name(str) - Name of the anime/series.
- limit(Optional[int]) - The maximum size of the list. The default value is 100.
- offset(Optional[int]) - The list offset. The default value is 0.
- fields(Optional[List [str]]) - List of fields used to show more information about the anime. If is empty, the default fields are id, title and main_picture.

Raises:

- ValueError - The 'field' is not a valid field!

Returns:
List of animes in JSON format.

Return type:
str

2.2 Services

Services that are used in the library.

Methods
validateFields

services.**validateFields**(*fields*)

Validate the fields provided.

Parameters: - fields(List [str]) - List of fields.

Raises:

- ValueError - The 'field' is not a valid field!

Returns:
If the fields are valid or not.

Return type:
bool

2.3 Models

Models are classes that are received from MyAnimeList and are not meant to be created by the user of the library.

> **Danger:** The classes listed below are not intended to be created by users and are also read-only.
>
> For example, this means that you should not make your own *Anime* instances nor should you modify the *Anime* instance yourself.

2.3.1 AlternativeTitles

class models.alternativeTitles.**AlternativeTitles**(*synonyms*, *english*, *japanese*)

Methods
getSynonyms()
getEnglish()
getJapanese()

Parameters:

- synonyms (List [str]) - A list of title synonyms.
- english (str) - English version of the title.
- japanese (str) - Japanese version of the title.

Methods

getSynonyms()

List of synonyms.

Return type:
List [str]

getEnglish()

English version.

Return type:
str

`getJapanese()`

Japanese version.

Return type:
str

2.3.2 Anime

`class models.anime.Anime(node, fields)`

Methods	Methods
getId()	*getTitle()*
getMainPicture	*getStartDate()*
getAlternativeTitle()	*getSynopsis()*
getEndDate()	*getRank()*
getMean()	*getNumUserList()*
getPopularity()	*getNsfwClassification()*
getNumScoringUsers()	*getCreatedAt()*
getGenres()	*getMediaType()*
getUpdatedAt()	*getNumEpisodes()*
getStatus()	*getBroadcast()*
getStartSeason()	*getAvgEpisodeDurationInSeconds()*
getSource()	*getStudios()*
getRating()	*getBackground()*
getPictures()	*getRelatedMangas()*
getRelatedAnimes()	*getStatistics()*
getRecommendations()	

Parameters:

- node (dict) - The JSON object anime.
- fields (List [str]) - The fields used for the request.

Methods

`getId()`

Anime ID.

Return type:
int

`getTitle()`

Anime title.

Return type:
str

`getMainPicture()`

Anime main picture.

Return type:
Picture

`getAlternativeTitle()`

The alternative title of the anime.

Return type:
AlternativeTitles

`getStartDate()`

The anime start date. Format `YYYY-mm-dd`.

Return type:
str

`getEndDate()`

The anime end date. Format `YYYY-mm-dd`.

Return type:
str

`getSynopsis()`

Anime synopsis.

Return type:
str

`getMean()`

Mean score.

Return type:
float

`getRank()`

Anime rank.

Return type:
int

`getPopularity()`

Anime popularity.

Return type:
int

`getNumUserList()`

The number of users who have the anime in their list.

Return type:
int

`getNumScoringUsers()`

The number of users who rated the anime.

Return type:
int

`getNsfwClassification()`

Anime NSFW classification.

Return type:
int

`getGenres()`

The list of anime genres.

Return type:
List [*Genre*]

`getCreatedAt()`

Timestamp of anime creation in MyAnimeList database.

Return type:
str

`getUpdatedAt()`

Timestamp of anime update in MyAnimeList database.

Return type:
str

`getMediaType()`

Anime media type.

Return type:
str

`getStatus()`

Airing status.

Return type:
str

`getNumEpisodes()`

The total number of episodes of this series. If unknown, it is 0.

Return type:
int

`getStartSeason()`

Anime start season.

Return type:
Season

`getBroadcast()`

Broadcast day of the week and start time (JST).

Return type:
Broadcast | None

`getSource()`

Original work.

Return type:
str

`getAvgEpisodeDurationInSeconds()`

Average length of episode in seconds.

Return type:
int

`getRating()`

Anime rating.

Return type:
str

`getStudios()`

List of studios that produced the anime.

Return type:
List [*Studio*]

`getPictures()`

List of anime pictures.

Important: You cannot contain this field in a list.

Return type:
List [*Picture*] | None

`getBackground()`

The API strips BBCode tags from the result.

Important: You cannot contain this field in a list.

Return type:
List [str] | None

`getRelatedAnimes()`

List of related animes.

Important: You cannot contain this field in a list.

Return type:
List [*RelatedNode*] | None

getRelatedMangas()

List of related mangas.

Important: You cannot contain this field in a list.

Return type:
List [*RelatedNode*] | None

getRecommendations()

Summary of recommended anime for those who like this anime.

Important: You cannot contain this field in a list.

Return type:
List [*Recommendation*] | None

getStatistics()

Anime statistics on MyAnimeList.

Important: You cannot contain this field in a list.

Return type:
List [*Statistics*] | None

2.3.3 Broadcast

class models.broadcast.**Broadcast**(*day_of_the_week*, *start_time*)

Methods
getDayOfTheWeek()
getStartTime()

Parameters:

- day_of_the_week (*DayWeekEnum*) - Day of the week broadcast in Japan time.
- start_time (str) - Time in hours format that is broadcasted.

Methods

getDayOfTheWeek()

Broadcast day of the week.

Return type:
str

getStartTime()

Anime start time in JST.

Return type:
str

2.3.4 Genre

class models.genre.**Genre**(*id*, *name*)

Methods
getId()
getName()

Parameters:

- id (int) - ID of the genre.
- name (str) - Name of the genre.

Methods

getId()

Genre ID.

Return type:
int

getName()

Genre name.

Return type:
str

2.3.5 Node

class models.node.**Node**(*id*, *title*, *main_picture*)

Methods
getId()
getTitle()
getMainPicture()

Parameters:

- id (int) -
- title (str) -
- main_picture () -

Methods

getId()

Anime or manga ID.

Return type:
int

getTitle()

Anime or manga title.

Return type:
str

getMainPicture()

Anime or manga main picture.

Return type:
Picture

2.3.6 Picture

class models.picture.**Picture**(*large*, *medium*)

Methods
getLarge()
getMedium()

Parameters:

- large (str) - The URI of an anime's large picture.
- medium (str) - The URI of an anime's medium picture.

Methods

getLarge()

Large size picture.

Return type:
str

getMedium()

Medium size picture.

Return type:
str

2.3.7 Recommendation

class models.recommendation.**Recommendation**(*id*, *title*, *main_picture*, *num_recommendations*)

Methods
getId()
getTitle()
getMainPicture()
getNumRecommendations()

Parameters:

- id (int) - ID of the anime.
- title (str) - Title of the anime.
- main_picture (*Picture*) - Main picture of the anime.
- num_recommendations (int) - Number of recommendations of the anime.

Methods

getId()

Anime ID.

Return type:
int

getTitle()

Anime title.

Return type:
str

getMainPicture()

Anime main picture.

Return type:
Picture

getNumRecommendations()

Number of recommendations.

Return type:
int

2.3.8 RelatedNode

class models.relatedNode.**RelatedNode**(*id*, *title*, *main_picture*, *relation_type*)

Methods
getId()
getTitle()
getMainPicture()
getRelationType()

Parameters:

- id (int) - ID of the anime or manga.
- title (str) - Title of the anime or manga.
- main_picture (*Picture*) - Main picture of the anime or manga.
- relation_type (*RelationTypeEnum*) - Relation type of the anime or manga.

Methods

getId()

Anime or manga ID.

Return type:
int

getTitle()

Anime or manga title.

Return type:
str

getMainPicture()

Anime or manga main picture.

Return type:
Picture

getRelationType()

Anime or manga relation type.

Return type:
str

2.3.9 Season

class models.season.**Season**(*year*, *season*)

Methods
getYear()
getSeason()

Parameters:

- year (int) - Year of season.
- season (*SeasonEnum*) - Season name.

Methods

getYear()

Year of season.

Return type:
int

getSeason()

Season name.

Return type:
str

2.3.10 Statistics

class models.statistics.**Statistics**(*num_list_users*, *status*)

Methods
getNumUserLis()
getStatus()

Parameters:

- num_list_users (int) - Number of users who added the anime to their list.
- status (*StatisticsStatus*) - Users list status.

Methods

getNumUserLis()

The number of users who have the anime in their list.

Return type:
int

getStatus()

Anime status in the users list.

Return type:
StatisticsStatus

2.3.11 StatisticsStatus

class models.statisticsStatus.**StatisticsStatus**(*watching*, *completed*, *on_hold*, *dropped*, *plan_to_watch*)

Methods
getNumWatching()
getNumCompleted()
getNumOnHold()
getNumDropped()
getNumPlanToWatch()

Parameters:

- watching (int) - Number of users who are watching.
- completed (int) - Number of users who completed.
- on_hold (int) - Number of users who are on hold.
- dropped (int) - Number of users who dropped.
- plan_to_watch (int) - Number of users who are plan to watch.

Methods

getNumWatching()

The number of users who are watching the anime.

Return type:
int

getNumCompleted()

The number of users who are completed the anime.

Return type:
int

getNumOnHold()

The number of users who are waiting for the anime.

Return type:
int

getNumDropped()

The number of users who are dropped the anime.

Return type:
int

getNumPlanToWatch()

The number of users who plan to watch the anime.

Return type:
int

2.3.12 Studio

class models.studio.**Studio**(*id*, *name*)

Methods
getId()
getName()

Parameters:

- id (int) - ID of the Anime Studio.
- name (str) - Name of the Anime Studio.

Methods

getId()

Studio ID.

Return type:
int

getName()

Studio name.

Return type:
str

2.4 Enumerations

The library provides some enumerations to avoid the API from being stringly typed in case the strings change in the future.

All enumerations are subclasses of enum.Enum.

> **Warning:** The enumerations listed below are not intended to be used by users and are also read-only.

2.4.1 AiringStatusEnum

enum `enums.airingStatusEnum.`**`AiringStatusEnum`**

Key	Value
finished_airing	Finished airing
currently_airing	Currently airing
not_yet_aired	Not yet aired

2.4.2 DayWeekEnum

enum `enums.dayWeekEnum.`**`DayWeekEnum`**

Key	Value
sunday	0
monday	1
tuesday	2
wednesday	3
thursday	4
friday	5
saturday	6

2.4.3 FieldsEnum

enum `enums.fieldsEnum.`**`FieldsEnum`**

Key	Value	Key	Value
id	0	media_type	16
title	1	status	17
main_picture	2	num_episodes	18
alternative_title	3	start_season	19
start_date	4	broadcast	20
end_date	5	source	21
synopsis	6	average_episode_duration	22
mean	7	rating	23
rank	8	studios	24
popularity	9	pictures	25
num_list_users	10	background	26
num_scoring_users	11	related_anime	27
nsfw	12	related_manga	28
genres	13	recommendations	29
created_at	14	statistics	30
updated_at	15		

Note: The keys `pictures`, `background`, `related_anime`, `related_manga`, `recommendations`, and `statistics` cannot be in the **fields** parameter of *getAnimeList()*, *getAnimeListInDict()* and *getAnimeListInJSON()* methods.

2.4.4 MediaTypeEnum

enum enums.mediaTypeEnum.**MediaTypeEnum**

Key	Value
unknown	0
tv	1
ova	2
movie	3
special	4
ona	5
music	6

2.4.5 NsfwEnum

enum enums.nsfwEnum.**NsfwEnum**

Key	Value
white	This work is safe for work
gray	This work may be not safe for work
black	This work is not safe for work

2.4.6 RatingEnum

enum enums.ratingEnum.**RatingEnum**

Key	Value
g	All Age
pg	Children
pg_13	Teens 13 and Older
r	17+ (violence & profanity)
r_plus	Profanity & Mild Nudity
rx	Hentai

2.4.7 RelationTypeEnum

enum enums.relationTypeEnum.**RelationTypeEnum**

Key	Value
sequel	Sequel
prequel	Prequel
alternative_setting	Alternative Setting
alternative_version	Alternative Version
side_story	Side Story
parent_story	Parent Story
summary	Summary
full_story	Full Story

2.4.8 SeasonEnum

enum enums.seasonEnum.**SeasonEnum**

Key	Value
winter	Winter
spring	Spring
summer	Summer
fall	Fall

2.4.9 SourceEnum

enum enums.sourceEnum.**SourceEnum**

Key	Value
other	Other
original	Original
manga	Manga
four_koma_manga	4 Koma manga
web_manga	Web manga
digital_manga	Digital manga
novel	Novel
light_novel	Light novel
visual_novel	Visual novel
game	Game
card_game	Card game
book	Book
picture_book	Picture book
radio	Radio
music	Music

CHAPTER

THREE

INSTALLATION

How to install the myanimelistpy library.

```
pip install myanimelistpy
```

or

```
# Linux/macOS
python3 -m pip install -U myanimelistpy

# Windows
py -m pip install -U myanimelistpy
```

CHAPTER

FOUR

GETTING STARTED

Is this your first time using the library? This is the place to get started!

Click here to check how to use **myanimelispy** and start developing with us.

CHAPTER

FIVE

MANUALS

These pages go into great detail about everything the library can do.

- *Reference Manual*

CHAPTER

SIX

HELP

If you're having any issues or questions with the library, head over to our GitHub issues page and let us know.

INDEX

V