Anipy Documentation

Release 0.1

Damian Maggio Esne

Jan 10, 2018

Contents

Anipy is a python library that wraps and organize the Anilist rest api into modules, classes and functions so it can be used quick, easy, and right out of the box. You can take a look at the api official docs. Anilist is a Josh Star's project.

Contents:

CHAPTER 1

Getting Started

1.1 Installation

For now the only available versions are alphas. You can Instaled the las by:

```
$ git clone https://github.com/twissell-/anipy.git
$ cd anipy
$ python setup.py # Be sure using Python 3
```

1.2 Usage

I've tried to keep the developer interface as simple as possible.

1.2.1 Authentication

Before you can access any Anilist resource you have to get authenticated. Once you have created a client you must configure `auth.AuthenticationProvider` class with your credentials.

Now you can get authenticated with any of the available grant types. Aditionaly, Anipy have a `GrantType.refreshToken` in case you have saved a refresh token from a previous authentication. *Note that only code and pin authentication gives you a refresh token.*

```
from anipy import AuthenticationProvider
from anipy import Authentication
from anipy import GrantType

AuthenticationProvider.config('your-client-id', 'your-client-secret', 'your-redirect-
↪uri')

auth = Authentication.fromCredentials()
# or
```

```
auth = Authentication.fromCode('code')
# or
auth = Authentication.fromPin('pin')

# Now you can save the refresh token
refresh_token = auth.refreshToken

auth = Authentication.fromRefreshToken(refresh_token)
```

Authentication expires after one hour and will refresh automatically, nevertheless you can do it manually at any time, ie.:

```
if auth.isExpired:
    auth.refresh()
```

1.2.2 Resources

Resources are one of the most important parts of the library. They are in charge of go an get the data from the Anilist API. Each domain class have a resource, you can compare them to *Data Access Objects*. All resouces are **Singletons**.

In order to keep things simple you can access the resource from class it serves

```
# Current logged user
user = User.resource().principal()
# A user for his Id or Display Name
user = User.resource().byId(3225)
user = User.resource().byDisplayName('demo')
```

Some resources are injected in other classes also in order to keep things simple (ie. `AnimeListResource`). So if you want to get de watching list of a user you can do:

```
# The long way
resource = AnimeListResource()
watching_list = resource.byUserId(user.id)
# Or the short way
watching_list = user.watching
```

CHAPTER 2

Indices and tables

- genindex
- modindex
- search