marvelous Documentation

Release 0.0.4

Robert Kuykendall

November 07, 2016

Contents

Links:

- Code on Github
- Published on PyPi
- Read the project documentation

To install:

```
pip install marvelous
```

Usage:

```
import marvelous

# Your own config file to keep your private key local and secret
from config import public_key, private_key

# Authenticate with Marvel, with keys I got from http://developer.marvel.com/
m = marvelous.api(public_key, private_key)

# Get all comics from this week, sorted alphabetically by title
pulls = sorted(m.comics({
    'format': "comic",
    'formatType': "comic",
    'noVariants': True,
    'dateDescriptor': "thisWeek",
    'limit': 100}),
    key=lambda comic: comic.title)

for comic in pulls:
    # Write a line to the file with the name of the issue, and the
    # id of the series
    print('{} (series #{})'.format(comic.title, comic.series.id))
```

CHAPTER 1

Introduction

This package uses 2 major packages :

- requests for connecting to the Marvel API
- marshmallow for parsing the response into python objects

marvelous was created to make it easy to write small but power scripts which leverage the Marvel API, such as reading lists. The original use was to automate a list of Marvel comics weekly releases without having to copy each title manually from the Midtown Comics website.

CHAPTER 2

Instantiating API

```
import marvelous

public_key = "Public key from https://developer.marvel.com/account"
private_key = "Private key from https://developer.marvel.com/account"

# Optional
class CacheClass:
    def get(self, key):
        # This method should return cahed value with key
        return None

    def store(self, key, value):
        # This method should store key value pair
        return

cache = CacheClass()

# m is a session object, read about it below
m = marvelous.api(
    public_key,
    private_key,
    cache=cache
)
```

CHAPTER 3

Session Methods

The session object returned by `marvelous.api(public_key, private_key)` has the following methods.

3.1 m.call(endpoint, params)

Useful for endpoints not covered below, complex API calls. Useful if you want to interact with the exact response from the API, but still want a library to help making calls earlier. For example:

```
api_response = m.call(
  ['characters', character_id, 'comics'],
  {'dateDescriptor': 'thisMonth', 'limit': 50 })
```

3.2 m.comics(params=None)

Calls the */v1/public/comics* endpoint with any params passed in and returns a *ComicList* object. For documentation on *ComicList*, see below. For documentation on all the params argument, see the Marvel API documentation

3.3 m.series(series_id)

Calls the */v1/public/series/{seriesId}* endpoint with the first argument and returns a *Series* object.

CHAPTER 4

Objects

Aside from making it easier to build and send requests, marvelous also creates east to work with python objects from the response.

4.1 Comic

- `id` - Int
- `digital_id` - Int, *digitalId* from API
- `title` - String
- `issue_number`- Int, *issueNumber* from API
- `variant_description` - String, *variantDescription* from API
- `description` - String
- `modified` - Datetime
- `isbn` - String
- `up` - String
- `diamond_code` - String, *diamondCode* from API
- `ean` - String
- `issn` - String
- `format` - String
- `page_count` - Int, *pageCount* from API
- `series` - `Series` object
- `dates` - `'Dates` object

Not yet implemented:

- `textObjects`
- `resourceURI`
- `urls`
- `variants`
- `collections`

- `collectedIssues`

4.2 ComicsList

- `comics` - List, `Comic` objects
- `response` - Dictionary, raw response body

4.3 Dates

- `on_sale` - Date, on sale
- `foc` - Date, Final Order Cut-off
- `unlimited` - Date, Marvel Unlimited

4.4 Series

- `response` - Dictionary, raw response body
- `id` - Int
- `resource_uri` - String, *resourceURI* from API
- `title` - String
- `comics` - Method, Returns `ComicsList` object for */v1/public/series/{seriesId}/comics*

CHAPTER 5

Exceptions

Exceptions can be imported from `marvelous.exceptions` and caught:

- `ApiError`
- `AuthenticationError`
- `CacheError`

CHAPTER 6

Caching

`marvelous.api` supports an optional cache attribute, which can store API responses and significanly improve working with the Marvel API. The cache argument must be an object with these two methods:

- `get(self, key):`
- `store(self, key, value):`

Anything else is up to the user. A cache could be implemented as a simple attribute in memoru or with the help of a database, redis, an API, files, or anything else.

An Sqlite cache class is included with the library and can be imported like so:

```
m = marvelous.api(
    public_key, private_key,
    cache=marvelous.SqliteCache(db_name="marvelous_cache.db"))
```

CHAPTER 7

Examples

```
import os
import marvelous

# Your own config file to keep your private key local and secret
from config import public_key, private_key

# All the series IDs of comics I'm not interested in reading
# I pull these out of the resulting pulls.txt file, then rerun this script
IGNORE = set([
    19709, 20256, 19379, 19062, 19486, 19242, 19371, 19210, 20930, 21328,
    20834, 18826, 20933, 20365, 20928, 21129, 20786, 21402, 21018
])

# Authenticate with Marvel, with keys I got from http://developer.marvel.com/
m = marvelous.api(public_key, private_key)

# Get all comics from this week, sorted alphabetically by title
# Uses the same API parameters as listed in the official API documentation
pulls = sorted(m.comics({
    'format': "comic",
    'formatType': "comic",
    'noVariants': True,
    'dateDescriptor': "thisWeek",
    'limit': 100}),
    key=lambda comic: comic.title)

# Grab the sale date of any of the comics for the current week
week = pulls[0].dates.on_sale.strftime('%m/%d')

print("New comics for the week of {}:".format(week))
# Check each comic that came out this week
for comic in pulls:
    # If this series isn't in my ignore list
    if comic.series.id not in IGNORE:
        # Write a line to the file with the name of the issue, and the
        # id of the series incase I want to add it to my ignore list
        print('- {} (series #{})'.format(comic.title, comic.series.id))
```

Example output:

```
New comics for the week of 11/09:
- All-New X-Men (2015) #15 (series #20622)
- Amazing Spider-Man: Renew Your Vows (2016) #1 (series #22545)
```

```
- Black Panther: World of Wakanda (2016) #1 (series #22549)
- Captain America: Steve Rogers (2016) #7 (series #21098)
- Daredevil (2015) #13 (series #20780)
- Dark Tower: The Drawing of the Three - The Sailor (2016) #2 (series #19377)
- Deadpool: Back in Black (2016) #3 (series #21489)
- Doctor Strange And The Sorcerers Supreme (2016) #2 (series #22560)
- Gwenpool (2016) #8 (series #21490)
- Han Solo (2016) #5 (series #19711)
- Invincible Iron Man (2016) #1 (series #22928)
- Max Ride: Final Flight (2016) #3 (series #22197)
- Mosaic (2016) #2 (series #20818)
- Ms. Marvel (2015) #13 (series #20615)
- Old Man Logan (2016) #13 (series #20617)
- Power Man and Iron Fist (2016) #10 (series #21122)
- Prowler (2016) #2 (series #22535)
- Solo (2016) #2 (series #22441)
- Spider-Gwen (2015) #14 (series #20505)
- Spider-Man/Deadpool (2016) #11 (series #19679)
- Star Wars: The Force Awakens Adaptation (2016) #6 (series #21493)
- The Avengers (2016) #1.1 (series #22966)
- The Clone Conspiracy (2016) #2 (series #22654)
- Thunderbolts (2016) #7 (series #20884)
- Uncanny Avengers (2015) #16 (series #20621)
- Uncanny X-Men (2016) #15 (series #20612)
```