Pykemon Documentation

Release 0.2.0

Paul Hallett

Dec 19, 2016

Contents

Contents:

CHAPTER 1

Pykemon

A python wrapper for PokeAPI

- Free software: BSD license
- Documentation: http://pykemon.rtfd.org.

1.1 Installation

Nice and simple:

```
$ pip install pykemon
```

1.2 Usage

Even simpler:

```
>>> import pykemon
>>> client = pykemon.V1Client()
>>> p = client.get_pokemon(uid=1)
[<Pokemon - Bulbasaur>]
```

1.3 Features

- Generate Python objects from PokeAPI resources.
- Human-friendly API

CHAPTER 2

Installation

At the command line:

```
$ easy_install pykemon
```

Or, if you have virtualenvwrapper installed:

```
$ mkvirtualenv pykemon
$ pip install pykemon
```

CHAPTER 3

Usage

To use Pykemon in a project:

```
>>> import pykemon
```

CHAPTER 4

New API

Since version 0.2.0, Pykemon now works with Beckett, an easy to use API Client Framework:

```
>>> client = pykemon.V1Client()
>>> bulba = client.get_pokemon(uid=1)[0]
<Pokemon | Bulbasaur>
>>> bulba.name
Bulbasaur
```

The following methods work with this client and all take a *uid* parameter:

- get_pokemon
- get_move
- get_sprite
- get_ability
- get_game
- get_type
- get_egg

CHAPTER 5

Version 0.1* API

Then you can start grabbing stuff from the API:

```
>>> pykemon.get(pokemon='mew')
<Pokemon - Mew>
>>> pykemon.get(pokemon_id=1)
<Pokemon - Bulbasaur>
```

Fully supports all the resources on PokeAPI:

```
>>> pykemon.get(move_id=15)
<Move - Cut>
>>> pykemon.get(ability_id=1)
<Ability - stench>
```

Resources that have other abilities linked are displayed as dicts:

```
>>> p = pykemon.get(pokemon_id=1)
>>> p
<Pokemon - Bulbasaur>
>>> p.evolutions
{'Ivysaur': '/api/v1/pokemon/2/'}
```

With the resource uri information you can request the linked resources easily.

Each resource is accessible, with it's own object-oriented representation. Every resource can be accessed with the term:

```
resource_id
```

Where 'resource' is replaced depending on the resource you want:

```
pokemon_id
move_id
ability_id
egg_id
type_id
description_id
game_id
sprite_id
```

The Pokemon resource can also be requested using the name:

```
>>> pykemon.get(pokemon='rotom')
<Pokemon - Rotom>
```

Make sure you use lower case strings!

CHAPTER 6

Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

6.1 Types of Contributions

6.1.1 Report Bugs

Report bugs at https://github.com/phalt/pykemon/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

6.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" is open to whoever wants to implement it.

6.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with "feature" is open to whoever wants to implement it.

6.1.4 Write Documentation

Pykemon could always use more documentation, whether as part of the official Pykemon docs, in docstrings, or even on the web in blog posts, articles, and such.

6.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/phalt/pykemon/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

6.2 Get Started!

Ready to contribute? Here's how to set up *pykemon* for local development.

1. Fork the *pykemon* repo on GitHub.
2. Clone your fork locally:

   ```
   $ git clone git@github.com:your_name_here/pykemon.git
   ```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

   ```
   $ mkvirtualenv pykemon
   $ cd pykemon/
   $ python setup.py develop
   ```

4. Create a branch for local development:

   ```
   $ git checkout -b name-of-your-bugfix-or-feature
   ```

   Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

   ```
   $ flake8 pykemon tests
   $ python setup.py test
   $ tox
   ```

   To get flake8 and tox, just pip install them into your virtualenv.

6. Commit your changes and push your branch to GitHub:

   ```
   $ git add .
   $ git commit -m "Your detailed description of your changes."
   $ git push origin name-of-your-bugfix-or-feature
   ```

7. Submit a pull request through the GitHub website.

6.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 2.6, 2.7, and 3.3, and for PyPy. Check https://travis-ci.org/phalt/pykemon/pull_requests and make sure that the tests pass for all supported Python versions.

6.4 Tips

To run a subset of tests:

```
$ python -m unittest tests.test_pykemon
```

CHAPTER 7

Credits

7.1 Development Lead

- Paul Hallett <paulandrewhallett@gmail.com>

7.2 Contributors

- Owen Hallett

Made a commit? Add your name to the list!

CHAPTER 8

History

8.1 0.2.0 (2016-06-11)

- Beckett API Client framework added

8.2 0.1.2 (2014-1-3)

- Sprite attribute added to Pokemon class

8.3 0.1.1 (2013-12-24)

- Description attribute added to Pokemon class

8.4 0.1.0 (2013-12-23)

- First release on PyPI.
- All PokeAPI resources fully supported and represented in an object-oriented style.
- Easy-to-use API: just one method!

CHAPTER 9

Indices and tables

- genindex
- modindex
- search