mkm-sdk

Release 0.3.0

February 13, 2016

Contents

A simple SDK for dedicated apps for Magic Kard Market.

Contents:

CHAPTER 1

Installation

From the command line:

```
pip install mkmsdk
```

For the SDK to work properly you need to create four environment variables holding the tokens necessary to create the authorization to make requests. You can find them in your Magic Kard Market account page under the apps section.

- MKM_APP_TOKEN
- MKM_APP_SECRET
- MKM_ACCESS_TOKEN
- MKM_ACCESS_TOKEN_SECRET

CHAPTER 2

Usage

To use the SDK the first thing to do is import *mkm* to work on live servers or *mkm_sandbox* to work on the sandbox:

```
from mkmsdk.mkm import mkm

from mkmsdk.mkm import mkm_sandbox
```

For example to obtain informations about the authenticated account you can make a request like this:

```
response = mkm.account_management.account()
```

This will return a Response object that contains the response from the server.

If you want get the content of the response you call the *json* method on the *response* object:

```
response.json()

{'account': {'name': {'firstName': 'Luke' 'lastName': 'Skywalker'},
  'country': 'DE',
  'isCommercial': 0,
  'riskGroup': 1,
  'bankRecharge': 0,
  'idUser': 123456,
  'sellCount': 0,
  'paypalRecharge': 0,
  'shipsFast': 0,
  'unreadMessages': 0,
  'username': 'SkyWalker',
  'onVacation': False,
  'reputation': 0,
  'articlesInShoppingCart': 0,
  'address': {'name': 'Luke Skywalker',
   'extra': '',
   'country': 'DE',
   'zip': '12345',
   'street': 'Tatooine',
   'city': 'Tatooine'},
  'idDisplayLanguage': 5,
  'accountBalance': 0}}
```

Similarly to obtain informations about a specific user you can make a request like this:

```
response = mkm.market_place.user(user='SampleUser')
```

CHAPTER 3

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:

3.1 Types of Contributions

3.1.1 Report Bugs

Report bugs at https://github.com/evonove/mkm-sdk/issues

3.1.2 Fix Bugs

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

3.1.3 Implement Features

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

3.1.4 Write Documentation

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

3.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/evonove/mkm-sdk/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 :)

  Get Started!

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

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

```
$ git clone git@github.com:your_name_here/mkmsdk.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 mkmsdk
$ cd mkmsdk/
$ 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 the tests, including testing other Python versions with tox:

```
$ python setup.py test
$ tox
```

To get tox, just pip install it 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.

3.2 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 2.7 and 3.4.

CHAPTER 4

History

4.1 0.3.0 (2016-02-13)

- Widget Apps now work correctly
- Fixed an issue when searching for cards with spaces, commas or other special chars in their name

4.2 0.2.0 (2015-07-30)

- Reworked how environment variables are retrieved
- Fixed requirements
- Fixed issue #1: HISTORY.rst was not found when installing the package

4.3 0.1.0 (2015-02-16)

- First release on PyPI