qosic-sdk Documentation

Release 5.0.1

Tobi DEGNON

Apr 17, 2023

CONTENTS:

CHAPTER

ONE

QOSIC-SDK

An unofficial python sdk for the QosIC platform. This platform provides an api to enable mobile money payments for businesses in Africa.

- Free software: MIT license
- Documentation: https://qosic-sdk.readthedocs.io.

1.1 Features

- Simple synchronous client to make your payment requests
- Cover 100% of Qosic public api
- Clean and meaningful exceptions
- 100 % test coverage
- Configurable timeouts

1.2 Quickstart

For those of you in a hurry, here's a sample code to get you started.

```
pip install qosic-sdk
```

```
from dotenv import dotenv_values
from qosic import Client, bj

config = dotenv_values(".env")

moov_client_id = config.get("MOOV_CLIENT_ID")
mtn_client_id = config.get("MTN_CLIENT_ID")
```

(continues on next page)

(continued from previous page)

```
login = config.get("SERVER_LOGIN")
password = config.get("SERVER_PASSWORD")

def main():
    phone = "229XXXXXXXX"
    mobile_carriers = [bj.MTN(id=mtn_client_id), bj.MOOV(id=moov_client_id)]
    client = Client(login=login, password=password, mobile_carriers=mobile_carriers)

    result = client.pay(phone=phone, amount=500)
    print(result)
    if result.success:
        print(f"Everything went fine")

    result = client.refund(reference=result.reference, phone=phone)
    print(result)


if __name__ == "__main__":
    main()
```

1.3 Credits

This package was created with Cookiecutter and the audreyr/cookiecutter-pypackage project template.

CHAPTER

TWO

INSTALLATION

2.1 Stable release

To install qosic-sdk, run this command in your terminal:

```
$ pip install qosic-sdk
```

This is the preferred method to install qosic-sdk, as it will always install the most recent stable release.

If you don't have pip installed, this Python installation guide can guide you through the process.

2.2 From sources

The sources for qosic-sdk can be downloaded from the Github repo.

You can either clone the public repository:

```
$ git clone git://github.com/Tobi-De/qosic-sdk
```

Or download the tarball:

```
$ curl -OJL https://github.com/Tobi-De/qosic-sdk/tarball/main
```

Once you have a copy of the source, you can install it with:

```
$ python setup.py install
```

CHAPTER

THREE

USAGE GUIDE

3.1 Overview

The `Client` class serves as the primary interface for communication with the QosIc API within the **qosic-sdk** package. This guide offers comprehensive instructions on how to effectively utilize the `Client` class for interacting with the payment platform API.

3.2 Initialization

To use the `Client` class, you need to initialize an instance of it with the required parameters. The following parameters are available for initialization:

- **login**: Your server authentication login/user.
- **password**: Your server authentication password.
- **mobile_carriers**: The list of configured mobile carriers.
- **base_url**: The QosIC server root domain, if you ever need to change it. Default is https://qosic.net:8443.
- **logger**: Custom logger for logging API requests and responses.

```
from qosic import Client, bj

mobile_carriers = [bj.MTN("mtn_client_id"), bj.MOOV("moov_client_id")]
client = Client(login="your_login", password="your_password", mobile_carriers=mobile_
↪carriers)
```

3.3 Making Payments

To make a payment using the `Client` class, you can call the `pay()` method with the following parameters:

- **phone**: The phone number of the payer.
- **amount**: The amount to be paid.
- **first_name** (optional): The first name of the payer. Default is an empty string.
- **last_name** (optional): The last name of the payer. Default is an empty string.

```
result = client.pay(phone="22901020304", amount=1000, first_name="John", last_name="Doe")
if result.success:
    print("Payment successful")
else:
    print("Payment failed. Error: ", result.response)
```

3.4 Processing Refunds

To request a refund using the `Client` class, you can call the `refund()` method with the following parameters:

- **reference**: The reference ID of the original payment, available in the `Result` object returned by the `pay()` method.
- **phone**: The phone number of the payer.

```
result = client.refund(reference="1234567890", phone="22901020304")
if result.success:
    print("Refund successful")
else:
    print("Refund failed. Error: ", result.response)
```

3.5 `Result` class

A helper class that encapsulates the response from the server for a payment or refund request made using the Python SDK for the Payment Platform API.

- **status** (Result.Status): The status of the request, which can be either `Result.Status.CONFIRMED` or `Result.Status.FAILED`.
- **reference** (str): The reference number associated with the request.
- **phone** (str): The phone number associated with the request.
- **mobile_carrier** (MobileCarrier): The mobile carrier associated with the request.
- **response** (httpx.Response): The HTTP response object returned from the server.

Properties

- **success** (bool): A property that indicates whether the request was successful or not. Returns `True` if the `status` is `Result.Status.CONFIRMED`, indicating a successful request, and `False` otherwise.

3.6 Logging

The `Client` class uses the `logging` module to log API requests and responses. You can customize the logging behavior and format by passing a custom logger to the `Client` class during initialization. For debugging purpose you can configure the logging level to display all the interactions with the payment API.

```
import logging

logging.basicConfig(level=logging.DEBUG)
```

(continues on next page)

(continued from previous page)

```
client = Client(login="your_login", password="your_password", mobile_carriers=mobile_
↪carriers)
client.pay(phone="22901020304", amount=1000) # will log everything in your terminal
```

3.7 Error Handling

If everything goes well, the `pay()` and `refund()` methods return a `Result` but if the request completely fails to be processed by the server, one the exceptions listed below is raised.

- **ServerError** : raised when the qos server is busy or fails for some reason.
- **UserAccountNotFoundError** : raised when the phone number provided does not have a mobile money account.
- **ProviderNotFoundError** : raised when for the given phone number, the provider can't be identified.
- **InvalidPhoneNumberError** : raised when the phone number does not match the valid format.
- **InvalidClientIDError** : raised when the client ID does not match the provider or is incorrect.
- **InvalidCredentialsError** : raised when your api credentials are invalid.

3.8 Best Practices

3.8.1 Use a task queue

To optimize the processing of payment requests, it's important to note that API response times may vary depending on the mobile carrier used. Factors such as waiting for customer approval or polling for transaction results can cause delays. To prevent your server from being blocked during this process, it's recommended to implement asynchronous processing using a task queue.

When using a task queue, you can periodically poll a specific endpoint on your server to check the status of the transaction. Alternatively, you can utilize server-side events to push real-time transaction status updates to the frontend. This allows you to efficiently display the transaction status to your users on the frontend, ensuring a smooth user experience.

3.8.2 Use environment variables for your credentials

You should never hardcode your credentials in your code. Instead, you should use environment variables to store your credentials and then access them in your code. This includes your **login**, **password**, and the **client_id** for all the mobile carriers you configured.

CHAPTER

FOUR

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:

4.1 Types of Contributions

4.1.1 Report Bugs

Report bugs at https://github.com/Tobi-De/qosic-sdk/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.

4.1.2 Fix Bugs

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

4.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with "enhancement" and "help wanted" is open to whoever wants to implement it.

4.1.4 Write Documentation

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

4.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/Tobi-De/qosic-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 :)

4.2 Get Started!

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

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

```
$ git clone git@github.com:your_name_here/qosic.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 qosic
$ cd qosic/
$ 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 qosic tests
$ python setup.py test or pytest
$ 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.

4.3 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 3.5, 3.6, 3.7 and 3.8, and for PyPy. Check https://travis-ci.com/Tobi-De/qosic-sdk/pull_requests and make sure that the tests pass for all supported Python versions.

4.4 Tips

To run a subset of tests:

```
$ pytest tests.test_qosic
```

4.5 Deploying

A reminder for the maintainers on how to deploy. Make sure all your changes are committed (including an entry in HISTORY.rst). Then run:

```
$ bump2version patch # possible: major / minor / patch
$ git push
$ git push --tags
```

Travis will then deploy to PyPI if tests pass.

CHAPTER

FIVE

CREDITS

5.1 Development Lead

- Tobi DEGNON <tobidegnon@proton.me>

5.2 Contributors

None yet. Why not be the first?

CHAPTER

SIX

HISTORY

6.1 3.0.1 (2022-03-29)

- Switch readme back to rst

6.2 3.0.1 (2022-03-29)

- Switch to poetry
- Add more tests

6.3 3.0.0 (2022-03-28)

- Complete rewrite
- Remove pydantic and phonenumbers as dependencies
- simplify the code

6.4 2.0.0 (2021-10-27)

- Now using polling2 and the phonenumbers python packages

6.5 1.1.3 (2021-05-22)

- add support for context manager to the Client class

6.6 1.1.2 (2021-05-20)

- Add more prefixes to MTN and MOOV prefixes
- Update docs

6.7 1.1.1 (2021-05-20)

- Update MTN and MOOV prefixes

6.8 1.1.0 (2021-05-19)

- Change MtnConfig step default and minimal value.
- Add exception list to the docs.
- Remove PollRuntimeError, now the poll function fails while raising the right exception.
- When active_logging is set to True, the client now collect responses in the client property *collected_responses*.

6.9 1.0.2 (2021-05-16)

- Change timeout defaults.
- Update docs.

6.10 1.0.1 (2021-05-14)

- Write some tests.
- The internal http client handles non-json responses better.

6.11 1.0.0 (2021-05-13)

- First release on PyPI.

CHAPTER

SEVEN

INDICES AND TABLES

- genindex
- modindex
- search