vat-validator

Afonso Silva

Aug 11, 2019

GETTING STARTED

Pythonic VAT validation library

CHAPTER

ONE

FEATURES

- Offline VAT code validation using country specific regular expressions and checksum algorithms;
- Online validation for European Union VAT codes using VIES web service;
- VAT code sanitization;
- Fully annotated with type hints, for a better IDE and `mypy` development experience;
- Tested and validated against 1697 different VAT codes.

CHAPTER

TWO

GETTING STARTED

2.1 Installation

`vat-validator` is distributed as standard pip library, and can be installed by running:

```
pip install vat-validator
```

To install the latest development version directly from git:

```
pip install git+git://github.com/ajcerejeira/vat-validator.git
```

2.2 Usage

```
>>> from vat_validator import inspect_vat, sanitize_vat, vat_is_valid
>>> vat_is_valid('PT', 'PT 502 011 378')
True
>>> sanitize_vat('PT', 'PT 502 011 378')
'502011378'
```

To validate a VAT number with `VIES` web service:

```
>>> from vat_validator.vies import check_vat
>>> check_vat('PT', '502 011 378')
CheckVATResult(country_code='PT', vat='502011378', request_date=datetime.date(2019, 6,
↪ 8), valid=True, name='UNIVERSIDADE DO MINHO', address='LG DO PACO\nBRAGA\n4700-320␣
↪BRAGA')
```

CHAPTER

THREE

ROADMAP

These are the goals before the *1.0.0* release:

- Have a comprehensive test suite with valid and invalid VAT codes for each country.
- Compare the validity of each VAT code used in tests with `TIN` web service.
- Remove `zeep` dependency, by using standard library `urllib` to make SOAP requests to VIES web service, making this a module without any dependencies.
- Add support for `async` requests to VIES web service.

CHAPTER

FOUR

CONTRIBUTING

Pull requests are welcome! Please check the *CONTRIBUTING* file for contribution guidelines.

CHAPTER

FIVE

LICENSE

This software is distributed under MIT license. See the *LICENSE* file for details.

CHAPTER

SIX

TABLE OF CONTENTS

6.1 Installation

`vat-validator` is distributed as standard pip library, and can be installed by running:

```
pip install vat-validator
```

To install the latest development version directly from git:

```
pip install git+git://github.com/ajcerejeira/vat-validator.git
```

6.2 Usage

```
>>> from vat_validator import inspect_vat, sanitize_vat, vat_is_valid
>>> vat_is_valid('PT', 'PT 502 011 378')
True
>>> sanitize_vat('PT', 'PT 502 011 378')
'502011378'
```

To validate a VAT number with `VIES` web service:

```
>>> from vat_validator.vies import check_vat
>>> check_vat('PT', '502 011 378')
CheckVATResult(country_code='PT', vat='502011378', request_date=datetime.date(2019, 6,
↪ 8), valid=True, name='UNIVERSIDADE DO MINHO', address='LG DO PACO\nBRAGA\n4700-320␣
↪BRAGA')
```

6.3 vat_validator.countries

vat_validator.countries.**EU_COUNTRY_CODES = {'AT', 'BE', 'BG', 'CY', 'CZ', 'DE', 'DK', 'EE',**

Set of European Union country code

6.4 vat_validator.inspectors

6.5 vat_validator.sanitizers

6.6 vat_validator.validators

This module contains a set of functions to validate a VAT number according to a country VAT format rules. Besides the general regex match, the validation functions perform some country specific calculations on the digits using algorithms such as MOD 11 or Lunh's algorithm.

All of these functions are in the format `vat_is_valid_XX` where XX is the IS0 3166 country code. They receive a string representing a VAT number in that country format and return `True` or `False` whether that code is valid or not.

Usage:

```
>>> vat_is_valid_pt('PT980405319')
True
>>> vat_is_valid_pt('PT-980 405 319')
True
>>> vat_is_valid_pt('980405319')
False
```

Each function is responsible to sanitize the input (remove preceding country code, spaces, punctuation, *etc.*. . .)

See also:

The list of VAT validation algorithms are published here: https://ec.europa.eu/taxation_customs/tin/

This wikipedia page contains a great overview of the different formats: https://en.wikipedia.org/wiki/VAT_identification_number

6.7 vat_validator.vies

This module allows interaction with VIES (VAT Information Exchange Service) web service. It can be used to validate VAT codes and fetch other information from the entity with that VAT.

Usage:

```
>>> check_vat('PT', '502011378')
CheckVATResult(country_code='PT',
               vat='502011378',
               request_date=2019-08-01 00:00:00+02:00,
               valid=True,
               name='UNIVERSIDADE DO MINHO',
               address='LG DO PACO\nBRAGA\n4700-320 BRAGA')
```

See also:

VIES on the European Comission website: http://ec.europa.eu/taxation_customs/vies/

The functions hereby described operate on this WSDL url: http://ec.europa.eu/taxation_customs/vies/checkVatService.wsdl

class vat_validator.vies.**CheckVATResult**(*country_code*, *vat*, *request_date*, *valid*, *name*, *address*)

Represents the result obtained by running the function *check_vat()*.

Parameters

- **country_code** (str) – IS0 3166 country code.
- **vat** (str) – VAT number.
- **request_date** (Optional[date]) – date when this result was queried.
- **valid** (bool) – whether this VAT is valid or not.
- **name** (Optional[str]) – optional name of the entity associated with this VAT.
- **address** (Optional[str]) – optional address of the entity associated with this VAT.

vat_validator.vies.**check_vat**(*country_code*, *vat*)

Checks if given VAT code is valid and (if possible) fetches the name and address of the entity with that code.

Parameters

- **country_code** (str) – valid IS0 3166 country code.
- **vat** (str) – VAT number to check.

Return type *CheckVATResult*

Returns instance of *CheckVATResult*.

Raises **ValueError** – when the country code is invalid or the VAT is empty.

6.8 Setting up

It is recommended to set up a virtual environment to develop on this project. To create one and install the required dependencies, run:

```
python -m venv venv
pip install -r requirements.txt
```

And you can start coding right ahead !

6.9 Documentation

The documentation is built with Sphinx documentation generator. To install Sphinx and the required plugins used in this project, run:

```
pip install -r docs/requirements.txt
```

To build the documentation:

```
cd docs/
make html    # or .\make.bat html if you are on a Windows platform
```

6.10 Testing

Currently the tests are implemented with standard Python unit testing framework `unittest`, and are stored in `tests/` directory.

To run the tests:

```
python -m unittest
```

6.11 Code style and linting

This project uses mypy, flake8 and black code style and formatter. This commands must return valid values on each commit:

```
mypy -m vat_validator
flake8 vat_validator tests
black -l 79 --check vat_validator/ tests/
```

It is recommended to set up your text editor/IDE to run this tools on save.

6.12 The MIT License

6.13 Indices and tables

- genindex
- modindex
- search

PYTHON MODULE INDEX

V

INDEX