vat-validator

Afonso Silva

Jun 13, 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 webservice;
- 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 Instalation

`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 validate_vat, countries_where_vat_is_valid
>>> validate_vat('PT', 'PT 502 011 378')
True
>>> countries_where_vat_is_valid('502 011 378')
['PT']
```

```
>>> 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

CONTRIBUTING

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

CHAPTER

FOUR

LICENSE

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

CHAPTER

FIVE

TABLE OF CONTENTS

5.1 Instalation

`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
```

5.2 Usage

```
>>> from vat_validator import validate_vat, countries_where_vat_is_valid
>>> validate_vat('PT', 'PT 502 011 378')
True
>>> countries_where_vat_is_valid('502 011 378')
['PT']
```

```
>>> 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')
```

5.3 vat_validator.vies

This module allows interaction with VIES (VAT Information Exchange Service) webservice. 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=datetime.date(2019, 6, 6),
               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** (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.

5.4 vat_validator.utils

vat_validator.utils.**countries_where_vat_is_valid**(*vat*)

Finds the countries where a given VAT code is valid.

Parameters **vat** (str) – VAT number to validate.

Return type List[str]

Returns list of country codes where the given VAT is valid.

vat_validator.utils.**sanitize_vat**(*vat*)

Sanitizes a given VAT code, removing the country prefix, whitespace and other non alphanumeric characters.

Parameters **vat** (str) – VAT code to sanitize.

Return type str

Returns sanitized VAT code.

Example

```
>>> sanitize_vat('PT 502.011.378')
'502011378'
```

vat_validator.utils.**validate_vat**(*country_code*, *vat*)

Validates a VAT number against the given country VAT format specification. It also takes care of sanitizing the VAT code, removing whitespaces and other invalid characters according to the country rules. It is not required for the VAT to start with the country code, so for example 'PT 980405319' and '980405319' will both yield the same result.

Parameters

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

Return type bool

Returns True if the given VAT is valid, False otherwise.

Raises **ValueError** – when the country code is invalid.

5.5 vat_validator.countries

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 validate_vat_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:

```
>>> validate_vat_pt('PT980405319')
True
>>> validate_vat_pt('PT-980 405 319')
True
>>> validate_vat_pt('80405319')
False
```

Each function is responsible to sanitize the input (remove preciding 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

vat_validator.countries.**EU_COUNTRY_CODES = ['AT', 'BE', 'BG', 'HR', 'CY', 'CZ', 'DE', 'DK',**

List of european union country codes

vat_validator.countries.**EU_RULES = {'AT': <function validate_vat_at>, 'BE': <function valid**

Maps a country code to the respective country VAT rule

vat_validator.countries.**validate_vat_at**(*vat*)

Validates a VAT number against austrian VAT format specification. In Austria is also named "Umsatzsteuer-Identifikationsnummer" (UID). The number must contain the letter 'U' followed by 8 digits and the last digit is the check digit.

Parameters **vat** (str) – VAT number to validate.

Return type bool

Returns True if the given VAT is valid, False otherwise.

vat_validator.countries.**validate_vat_be**(*vat*)
Validates a VAT number against belgian VAT format specification. In Belgium is also named "BTW identificatienummer" (BTW-nr). The number must contain 10 digits starting with 0 or 1. The old numbering schema had 9 digits, and is also accepted by this function.

Parameters **vat** (str) – VAT number to validate.

Return type bool

Returns True if the given VAT is valid, False otherwise.

vat_validator.countries.**validate_vat_bg**(*vat*)
Validates a VAT number against bulgarian VAT format specification. In Bulgary is also named "Identifikacionen nomer po DDS" ( ). The number must contain 9 or 10 digits. It assumes one of four formats: "legal entities", "physical persons", "foreigners" and "miscellaneous".

Parameters **vat** (str) – VAT number to validate.

Return type bool

Returns True if the given VAT is valid, False otherwise.

vat_validator.countries.**validate_vat_cy**(*vat*)
Validates a VAT number against cyprus VAT format specification. In Cyprus is also named "Arithmós Engraphs phi. pi. a." (). The number must contain 8 digits followed by a letter that is used to check the number.

Parameters **vat** (str) – VAT number to validate.

Return type bool

Returns True if the given VAT is valid, False otherwise.

vat_validator.countries.**validate_vat_cz**(*vat*)
Validates a VAT number against czech republic VAT format specification. In Czech Republic is also named "Daňové identifikační číslo" (DIČ). The number must contain 8 to 10 digits depending if it is a legar entity or individual.

Parameters **vat** (str) – VAT number to validate.

Return type bool

Returns True if the given VAT is valid, False otherwise.

vat_validator.countries.**validate_vat_de**(*vat*)
Validates a VAT number against german VAT format specification. In Germany is also named "Umsatzsteuer-Identifikationsnummer" (USt-IdNr). The number must contain 9 digits and the first one cannot be 0.

Parameters **vat** (str) – VAT number to validate.

Return type bool

Returns True if the given VAT is valid, False otherwise.

vat_validator.countries.**validate_vat_dk**(*vat*)
Validates a VAT number against dannish VAT format specification. In Denmark is also named "Momsregistreringsnummer" (CVR). The number must contain 8 digits and the last digit is the check digit. It uses MOD 11 algorithm to calculate the check digit.

Parameters **vat** (str) – VAT number to validate.

Return type bool

Returns True if the given VAT is valid, False otherwise.

See also:

https://erhvervsstyrelsen.dk/modulus_11

vat_validator.countries.**validate_vat_ee**(*vat*)
Validates a VAT number against estoniaon VAT format specification. In Estonia is also named "Käibemaksukohustuslase number" (KMKR). The number must contain 9 digits.

Parameters **vat** (str) – VAT number to validate.

Return type bool

Returns True if the given VAT is valid, False otherwise.

vat_validator.countries.**validate_vat_el**(*vat*)
Validates a VAT number against greece VAT format specification. In Greece is also named "Arithmós Forologikou Mētrou" (). The number must contain 9 digits and the last digit is the check digit. It uses MOD 11 algorithm to calculate the check digit.

Parameters **vat** (str) – VAT number to validate.

Return type bool

Returns True if the given VAT is valid, False otherwise.

vat_validator.countries.**validate_vat_fi**(*vat*)
Validates a VAT number against finnish VAT format specification. In Finland is also named "Arvonlisäveronumero" (AVL nro). The number must contain 8 digits and the last digit is the check digit. It uses MOD 11-2 algorithm to calculate the check digit.

Parameters **vat** (str) – VAT number to validate.

Return type bool

Returns True if the given VAT is valid, False otherwise.

See also:

http://tarkistusmerkit.teppovuori.fi/tarkmerk.htm#y-tunnus2

vat_validator.countries.**validate_vat_fr**(*vat*)
Validates a VAT number against french VAT format specification. In France is also named "Numéro de TVA intracommunautaire" (TVA). The number must contain 2 control characters followed by 9 digits.

Parameters **vat** (str) – VAT number to validate.

Return type bool

Returns True if the given VAT is valid, False otherwise.

vat_validator.countries.**validate_vat_hr**(*vat*)
Validates a VAT number against croatian VAT format specification. In Croatia is also named "PDV Id. Broj OIB" (PDV-ID; OIB). The number must contain 11 digits and the last digit is the check digit. It uses MOD 11-10 algorithm to calculate the check digit.

Parameters **vat** (str) – VAT number to validate.

Return type bool

Returns True if the given VAT is valid, False otherwise.

vat_validator.countries.**validate_vat_hu**(*vat*)
Validates a VAT number against hungarian VAT format specification. In Hungary is also named "Közösségi adószám" (NUM). The number must contain 8 digits and the last digit is the check digit.

Parameters **vat** (str) – VAT number to validate.

Return type bool

Returns True if the given VAT is valid, False otherwise.

vat_validator.countries.**validate_vat_ie**(*vat*)

Validates a VAT number against irish VAT format specification. In Ireland is also named "Value added tax identification no." (VAT/CBL). The number must be in one of the following formats:

- 7 digits and 1 letter, optionally followed by another letter
- 1 digit, 1 letter or "+", "*" and 5 digits and 1 letter

The number must contain 8 digits and the last digit is the check digit.

Parameters **vat** (str) – VAT number to validate.

Return type bool

Returns True if the given VAT is valid, False otherwise.

vat_validator.countries.**validate_vat_it**(*vat*)

Validates a VAT number against italien VAT format specification. In Italy is also named "Partita IVA" (P.IVA). The number must contain 11 digits and the last digit is the check digit. It uses Luhn Algorithm to calculate the check digit.

Parameters **vat** (str) – VAT number to validate.

Return type bool

Returns True if the given VAT is valid, False otherwise.

See also:

https://en.wikipedia.org/wiki/Luhn_algorithm

vat_validator.countries.**validate_vat_lt**(*vat*)

Validates a VAT number against lithuanian VAT format specification. In Lithuania is also named "Pridėtinės vertės mokestis" (PVM KODAS). The number must contain 9 or 12 digits.

Parameters **vat** (str) – VAT number to validate.

Return type bool

Returns True if the given VAT is valid, False otherwise.

vat_validator.countries.**validate_vat_lu**(*vat*)

Validates a VAT number against luxembourg VAT format specification. In Luxembourg is also named "Numéro d'identification à la taxe sur la valeur ajoutée" (No. TVA). The number must contain 8 digits and the last two digits are the check digits.

Parameters **vat** (str) – VAT number to validate.

Return type bool

Returns True if the given VAT is valid, False otherwise.

vat_validator.countries.**validate_vat_lv**(*vat*)

Validates a VAT number against latvian VAT format specification. In Latvia is also named "Pievienotās vērtības nodokļa" (PVN). The number must contain 11 digits and can be in one of the following formats:

- 1st digit is bigger than 3, and so use a MOD 11 algorithm
- 1st digit is 3, 2nd digit is 2, followed by 9 digits
- 2 digits represent a day of month, followed by 2 digits that represent the month, followed by 2 digits that represent the year, followed by 1 digit equals to 0, 1 or 2, followed by 4 digits

Parameters **vat** (str) – VAT number to validate.

Return type bool

Returns True if the given VAT is valid, False otherwise.

vat_validator.countries.**validate_vat_mt**(*vat*)
Validates a VAT number against maltese VAT format specification. In Malta is also named "Vat reg. no." (VAT No.). The number must contain 8 digits and the last two digits are the check digits. It uses MOD 37 algorithm to calculate the check digits.

Parameters **vat** (str) – VAT number to validate.

Return type bool

Returns True if the given VAT is valid, False otherwise.

vat_validator.countries.**validate_vat_nl**(*vat*)
Validates a VAT number against netherlands VAT format specification. In Netherlands is also named "Btw-nummer" Btw-nr.). The number must contain 9 digits followed by the letter 'B' followed by 2 digits. It uses MOD 11 algorithm to calculate the check digit.

Parameters **vat** (str) – VAT number to validate.

Return type bool

Returns True if the given VAT is valid, False otherwise.

vat_validator.countries.**validate_vat_pl**(*vat*)
Validates a VAT number against polish VAT format specification. In Poland is also named "numer identyfikacji podatkowej" (NIP). The number must contain 10 digits and the last digit is the check digit. It uses MOD 11 algorithm to calculate the check digit.

Parameters **vat** (str) – VAT number to validate.

Return type bool

Returns True if the given VAT is valid, False otherwise.

vat_validator.countries.**validate_vat_pt**(*vat*)
Validates a VAT number against portuguese VAT format specification. In Portugal is also named "Número de Identificação Fiscal" (NIF). The number must contain 9 digits and the last digit is the check digit. It uses MOD 11 algorithm to calculate the check digit.

Parameters **vat** (str) – VAT number to validate.

Return type bool

Returns True if the given VAT is valid, False otherwise.

See also:

https://pt.wikipedia.org/wiki/Número_de_identificação_fiscal

vat_validator.countries.**validate_vat_ro**(*vat*)
Validates a VAT number against romanian VAT format specification. In Romania is also named "Codul de identificare fiscală" (CIF). The number must contain 2 to 10 digits and the last digit is the check digit. It uses MOD 11 algorithm to calculate the check digit.

Parameters **vat** (str) – VAT number to validate.

Return type bool

Returns True if the given VAT is valid, False otherwise.

See also:

https://vatdesk.eu/en/romania/

vat_validator.countries.**validate_vat_se**(*vat*)
Validates a VAT number against swedish VAT format specification. In Sweden is also named "momsregistreringsnummer" (Momsnr). The number must contain 12 digits.

Parameters **vat** (str) – VAT number to validate.

Return type bool

Returns True if the given VAT is valid, False otherwise.

vat_validator.countries.**validate_vat_si**(*vat*)
Validates a VAT number against slovenian VAT format specification. In Slovenia is also named "Davčna številka" (ID za DDV). The number must contain 8 digits and the last digit is the check digit. It uses MOD 11 algorithm to calculate the check digit.

Parameters **vat** (str) – VAT number to validate.

Return type bool

Returns True if the given VAT is valid, False otherwise.

vat_validator.countries.**validate_vat_sk**(*vat*)
Validates a VAT number against slovakian VAT format specification. In Slovakia is also named "Identifikačné číslo pre daň z pridanej hodnoty" (IČ DPH). The number must contain 10 digits and be divisible by 11.

Parameters **vat** (str) – VAT number to validate.

Return type bool

Returns True if the given VAT is valid, False otherwise.

5.6 Setting up

It is recomended 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 !

5.7 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
```

5.8 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
```

5.9 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 recomended to set up your text editor/IDE to run this tools on save.

5.10 Indices and tables

- genindex
- modindex
- search

PYTHON MODULE INDEX

V

INDEX