
Translate Python Documentation

Release 3.5.0

Terry Yin, Rafael Henter

Nov 29, 2021

Contents

1 Quickstart	3
2 User Guide	5
3 Development	9
4 Other	11
5 Downloads	13

Translate is a simple but powerful translation tool written in python with with support for multiple translation providers. By now we are integrated with Microsoft Translation API and Translated MyMemory API

1.1 Installation

1.1.1 Requirements

Python 2.x, 3.x

To install via pip:

```
$ pip install translate
```

Or, you can download/clone the source and make manually:

```
$ git clone git@github.com:terryyin/translate-python.git
$ cd translate-python
$ python setup.py install
```


2.1 Overview

Translate is a simple but powerful translation tool written in python with with support for multiple translation providers. It can be used as python module or as command line tool

TODO: Add more content

2.2 Providers

Providers are responsible to translate the text

2.2.1 MicrosoftProvider

It is a paid provider but it is possible you can create a free account tends a quota of up to 2m of words per day

MicrosoftProvider (located at `translate.providers`) receives the following options:

`to_lang`, `from_lang='en'`, `secret_access_key`

- `to_lang`: language you want to translate
- `from_lang`: Language of the text being translated (optional): as default `autodetect`
- `secret_access_key`: OAuth Access Token

for further information about the provider:

<http://docs.microsofttranslator.com/> <https://azure.microsoft.com/en-us/pricing/details/cognitive-services/translator-text-api/>

2.2.2 MyMemory

Is a free provider but very complete

MyMemory (located at `translate.providers`) receives the following options:

`to_lang`, `from_lang='en'`, `email`

- `to_lang`: language you want to translate
- `from_lang`: Language of the text being translated (optional): as default `autodetect`
- `email`: Valid email to increase your translations cote

for further information about the provider:

<https://mymemory.translated.net/doc/spec.php> <http://mymemory.translated.net/doc/usagelimits.php>

2.2.3 LibreTranslate

Free and open source translation provider

LibreTranslate (located at `translate.providers`) receives the following options:

`to_lang`, `from_lang='en'`, `secret_access_key=None`, `base_url="https://translate.astian.org/"`

- `to_lang`: language you want to translate
- `from_lang`: Language of the text being translated (optional): as default `autodetect`
- **`secret_access_key`: LibreTranslate API key**
 - `base_url`: LibreTranslate instance url

for further information about the provider:

<https://libretranslate.com> <http://github.com/LibreTranslate/LibreTranslate>

2.3 Tutorial

2.3.1 Command Line

In your command-line:

```
$ translate-cli -t zh "This is a pen."
Translation:
-----
Translated by: MyMemory
```

Or

```
$ translate-cli -t zh "This is a pen." -o
```

Options

```
$ translate-cli --help
Usage: __main__.py [OPTIONS] TEXT...

Python command line tool to make on line translations

Example:

    $ translate-cli -t zh the book is on the table

Available languages:

    https://en.wikipedia.org/wiki/ISO_639-1
    Examples: (e.g. en, ja, ko, pt, zh, zh-TW, ...)
```

Options:

<code>--version</code>	Show the version and exit.
<code>--generate-config-file</code>	Generated the config file using a Wizard and exit.
<code>-f, --from TEXT</code>	Sets the language of the text being translated. The default value is 'autodetect'.
<code>-t, --to TEXT</code>	Sets the language you want to translate.
<code>-p, --provider TEXT</code>	Set the provider you want to use. The default value is 'mymemory'.
<code>--secret_access_key TEXT</code>	Set the secret access key used to get provider oAuth token.
<code>-o, --output_only</code>	Set to display the translation only.
<code>--help</code>	Show this message and exit.

Change Default Languages

In `~/python-translate.cfg`:

```
[DEFAULT]
from_lang = autodetect
to_lang = de
provider = mymemory
secret_access_key =
```

The `cfg` is not for using as a Python module.

or run the command line and follow the steps:

```
$ translate-cli --generate-config-file
Translate from [autodetect]:
Translate to: <language you want to translate>
Provider [mymemory]:
Secret Access Key []:
```

The country code, as far as I know, is following https://en.wikipedia.org/wiki/ISO_639-1.

2.3.2 Use As A Python Module

```
In [1]: from translate import Translator
In [2]: translator= Translator(to_lang="zh")
In [3]: translation = translator.translate("This is a pen.")
Out [3]:
```

The result is in translation, and it's usually a unicode string.

Use a different translation provider

```
In [1]: from translate import Translator
In [2]: to_lang = 'zh'
In [3]: secret = '<your secret from Microsoft>'
In [4]: translator = Translator(provider='microsoft', to_lang=to_lang, secret_access_
↳key=secret)
In [5]: translator.translate('the book is on the table')
Out [5]: ''
```

3.1 Development Installation

3.1.1 Requirements

Python 2.x, 3.x

3.1.2 Development install

After forking or checking out:

```
$ cd translate-python/  
$ pip install -r requirements-dev.txt  
$ pre-commit install
```

The requirements-dev are only used for development, so we can easily install/track dependencies required to run the tests using continuous integration platforms.

The official entrypoint for distribution is the `requirements.txt` which contains the minimum requirements to execute the tests.

Running tests:

```
$ make test
```

or:

```
$ py.test -vv -s
```

Generating documentation:

```
$ cd docs/  
$ make html
```

3.2 Release

To release a new version, a few steps are required:

- Update version/release number in `docs/source/conf.py`
- Add entry to `CHANGES.rst` and documentation
- Review changes in test requirements `requirements.txt`
- Test build with `make build`
- Commit changes
- Release with `make release`

4.1 Changelog

4.1.1 Changelog

4.1.2 3.6.1

- Add LibreTranslate

4.1.3 3.5.0

- Add sphinx documentation
- Update readme.

4.1.4 3.4.1

- Makefile: Add a make release command
- Add twine to dev requirements.

4.1.5 3.4.0

- Refactor: Create a folder to add all providers instead to let in a single file
- Add Microsoft provider
- Add more documentation to all providers (Translated-MyMemory and Microsoft Translator)
- Add arguments to change the default provider using translate-cli

4.1.6 3.3.0

- Refactor translate-cli (command line interface) Using Click library instead of ArgParser
- Unify translate-cli and main to avoid duplicate code
- Add documentation to be used on helper commands on translate-cli
- Remove unnecessary code
- Refactor setup to complete information in the PKG-INFO used by PyPI

4.1.7 3.2.1

- Change the license from “BEER-WARE” to MIT

4.1.8 3.2.0

- Add multiple providers suport

4.1.9 3.1.0

- Apply Solid Principles
- Organize Project
- Add pre-commit, pytest
- Add new Make file
- Add new test cases

4.1.10 3.0.0

- General Refactor
- Remove urllib to use requests
- Refactor methods names removing google from then
- Apply PEP8
- Change constructor to keep it the code simple

4.1.11 2.0.0 (2017-11-08)

- initial release using changes file

CHAPTER 5

Downloads

- PDF
- Epub