potranslator Documentation

Release 1.1.5

SekouD

Nov 01, 2018

Contents

Contents:

CHAPTER 1

potranslator

potranslator is a package to easily translate po and pot files generated by Sphinx or other tools in any language supported by Google Translate.

potranslator auto-detects the language in the original pot files and auto-translates the pot files into the supplied target languages.

The Command Line Interface of *potranslator* and its documentation are based on sphinx-intl.

Optional: supports the Transifex collaborative service to upload the auto-generated translations to transifex for collaborative corrections of the translations.

1.1 Supported Languages

- Afrikaans af
- Albanian sq
- Amharic am
- Arabic ar
- Armenian hy
- Azeerbaijani az
- Basque eu
- Belarusian be
- Bengali bn
- Bosnian bs
- Bulgarian bg
- Catalan ca
- Cebuano ceb (ISO-639-2)

- Chinese (Simplified) zh-CN (BCP-47)
- Chinese (Traditional) zh-TW (BCP-47)
- Corsican co
- Croatian hr
- Czech cs
- Danish da
- Dutch nl
- English en
- Esperanto eo
- Estonian et
- Finnish fi
- French fr
- Frisian fy
- Galician gl
- Georgian ka
- German de
- Greek el
- Gujarati gu
- Haitian Creole ht
- Hausa ha
- Hawaiian haw (ISO-639-2)
- Hebrew iw
- Hindi hi
- Hmong hmn (ISO-639-2)
- Hungarian hu
- Icelandic is
- Igbo ig
- Indonesian id
- Irish ga
- Italian it
- Japanese ja
- Javanese jw
- Kannada kn
- Kazakh kk
- Khmer km
- Korean ko

- Kurdish ku
- Kyrgyz ky
- Lao lo
- Latin la
- Latvian lv
- Lithuanian lt
- Luxembourgish lb
- Macedonian mk
- Malagasy mg
- Malay ms
- Malayalam ml
- Maltese mt
- Maori mi
- Marathi mr
- Mongolian mn
- Myanmar (Burmese) my
- Nepali ne
- Norwegian no
- Nyanja (Chichewa) ny
- Pashto ps
- Persian fa
- Polish pl
- Portuguese (Portugal, Brazil) pt
- Punjabi pa
- Romanian ro
- Russian ru
- Samoan sm
- Scots Gaelic gd
- Serbian sr
- Sesotho st
- Shona sn
- Sindhi sd
- Sinhala (Sinhalese) si
- Slovak sk
- Slovenian sl
- Somali so

- Spanish es
- Sundanese su
- Swahili sw
- Swedish sv
- Tagalog (Filipino) tl
- Tajik tg
- Tamil ta
- Telugu te
- Thai th
- Turkish tr
- Ukrainian uk
- Urdu ur
- Uzbek uz
- Vietnamese vi
- Welsh cy
- Xhosa xh
- Yiddish yi
- Yoruba yo
- Zulu zu

1.2 Quick Start for auto-translation with potranslator

This section describes how to translate documents generated by Sphinx with the *potranslator* command.

1. Create your document(s) by using Sphinx:

```
$ sphinx-build -b html /path/to/docs path/to/docs/_build
```

2. Optionally add the settings to your *conf.py* if you have one:

```
locale_dirs = ['locale/']   #path is an example but this is the␣
↪recommended path.
gettext_compact = False     #optional.
```

   locale_dirs is required and *gettext_compact* is optional.

3. Extract the document's translatable messages into pot files (make sure you are in the folder containing *make.bat* and *Makefile* if you are on windows):

```
$ make gettext
```

4. Translate/Update your documents in German and Japanese:

```
$ potranslator update -p _build/gettext -l de -l ja
```

   Done. You got these directories that contain po files with auto-translated entries:

```
./locale/de/LC_MESSAGES/
./locale/ja/LC_MESSAGES/
```

5. Translate/Update your documents in Japanese, build the compiled mo files and generate the translated html documents:

   Command line (for Unix systems):

   ```
   $ potranslator build
   $ make -e SPHINXOPTS="-D language='ja'" html
   ```

   Command line (for Windows cmd.exe):

   ```
   > set SPHINXOPTS=-D language=de
   > potranslator build
   > .\make.bat html
   ```

   Command line (for PowerShell):

   ```
   > Set-Item env:SPHINXOPTS "-D language=de"
   > potranslator build
   > .\make.bat html
   ```

That's all!

1.3 Basic Features

- Translate from pot files or update existing po files with auto-generated translation.
- Build mo files from translated po or pot files.

1.3.1 Requirements for the basic features

- Python 3.6, 3.5, 3.4, 2.7, pypy.
- external libraries: setuptools, six, babel, click, googletrans, polib

1.4 Optional features

These features need the transifex-client library.

- create a .transifexrc file from an environment variable, without interactive input.
- create a .tx/config file without interactive input.
- update a .tx/config file from locale/pot files automatically.
- build mo files from po files in the locale directory.

You need to use the *tx* command to use the following features:

- *tx push -s* : push pot (translation catalogs) to transifex.
- *tx pull -l ja* : pull po (translated catalogs) from transifex.

1.4.1 Requirements for the optional features

- Your transifex account if you want to upload/download po files from transifex.
- external library: transifex-client

1.5 Installation

It is strongly recommended to use virtualenv for this procedure:

```
$ pip install potranslator
```

If you want to use the *Optional Features*, you need install this additional library:

```
$ pip install potranslator[transifex]
```

1.6 Commands, options, environment variables

1.6.1 Commands

Type *potranslator* without arguments to show the help instructions.

1.6.2 Setup environment variables

All command-line options can be set with environment variables using the format POTRANSLATOR_<UPPER_LONG_NAME> .

Dashes (-) have to replaced with underscores (_).

For example, to set the target languages:

```
$ export POTRANSLATOR_LANGUAGE=de,ja
```

On the Windows command line:

```
> set POTRANSLATOR_LANGUAGE=de,ja
```

This is the same as passing the option to potranslator directly:

```
$ potranslator <command> --language=de --language=ja
```

1.6.3 Setup sphinx conf.py

Add the following settings to your sphinx document's conf.py if it exists:

```
locale_dirs = ['locale/']   #for example
gettext_compact = False     #optional
```

1.6.4 Setup Makefile / make.bat

make gettext will generate pot files into the *_build/gettext* directory, however pot files can be generated in the *locale/pot* directory if convenient.

You can do that by replacing *_build/gettext* with *locale/pot* in your *Makefile* and/or *make.bat* that was generated by sphinx-quickstart.

1.7 License

Licensed under the BSD license. See the LICENSE file for specific terms.

1.8 Original

The Command Line Interface and the transifex integration of *potranslator* are adapted from sphinx-intl.

- https://pypi.org/project/sphinx-intl

1.9 CHANGES

See: https://github.com/SekouD/potranslator/blob/master/HISTORY.rst

CHAPTER 2

Installation

2.1 Stable release

To install potranslator, run this command in your terminal:

```
$ pip install potranslator
```

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

If you want to use the Optional Features, you need to install this additional library transifex-client:

```
$ pip install potranslator[transifex]
```

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

2.2 From sources

The sources for potranslator can be downloaded from the Github repo.

You can either clone the public repository:

```
$ git clone git://github.com/SekouD/potranslator
```

Or download the tarball:

```
$ curl  -OL https://github.com/SekouD/potranslator/tarball/master
```

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

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

CHAPTER 3

Usage

3.1 From a Python program

To use potranslator in a python project:

```
from potranslator import PoTranslator

languages = ('fr', 'es', 'it')
translator = PoTranslator(pot_dir='path/to/pot_dir', locale_dir='path/to/locale_dir')

results = translator.translate_all_pot(src_lang='en', target_langs=languages, auto_
↪save=False)
```

3.2 Commands, options, environment variables

3.2.1 Commands

Type *potranslator* without arguments to show the help instructions.

3.2.2 Basic Usage

This section describes how to translate documents generated by Sphinx with the *potranslator* command.

1. Create your document(s) by using Sphinx:

   ```
   $ sphinx-build -b html /path/to/docs path/to/docs/_build
   ```

2. Optionally add the settings to your *conf.py* if you have one:

```
locale_dirs = ['locale/']   #path is an example but this is the␣
↪recommended path.
gettext_compact = False     #optional.
```

locale_dirs is required and *gettext_compact* is optional.

3. Extract the document's translatable messages into pot files (make sure you are in the folder containing *make.bat* and *Makefile* if you are on windows):

```
$ make gettext
```

4. Translate/Update your documents in German and Japanese:

```
$ potranslator update -p _build/gettext -l de -l ja
```

Done. You got these directories that contain po files with auto-translated entries:

```
./locale/de/LC_MESSAGES/
./locale/ja/LC_MESSAGES/
```

5. Translate/Update your documents in Japanese, build the compiled mo files and generate the translated html documents:

Command line (for Unix systems):

```
$ potranslator build
$ make -e SPHINXOPTS="-D language='ja'" html
```

Command line (for Windows cmd.exe):

```
> set SPHINXOPTS=-D language=de
> potranslator build
> .\make.bat html
```

Command line (for PowerShell):

```
> Set-Item env:SPHINXOPTS "-D language=de"
> potranslator build
> .\make.bat html
```

That's all!

3.2.3 Setup environment variables

All command-line options can be set with environment variables using the format POTRANSLATOR_<UPPER_LONG_NAME> .

Dashes (-) have to replaced with underscores (_).

For example, to set the languages:

```
$ export POTRANSLATOR_LANGUAGE=de,ja
```

On the Windows command line:

```
> set POTRANSLATOR_LANGUAGE=de,ja
```

This is the same as passing the option to potranslator directly:

```
$ potranslator <command> --language=de --language=ja
```

3.2.4 Setup sphinx conf.py

Add the following settings to your sphinx document's conf.py if it exists:

```
locale_dirs = ['locale/']   #for example
gettext_compact = False     #optional
```

3.2.5 Setup Makefile / make.bat

make gettext will generate pot files into *_build/gettext* directory, however pot files can be generated in the *locale/pot* if convenient.

You can do that by replacing *_build/gettext* with *locale/pot* in your *Makefile* and/or *make.bat* that was generated by sphinx-quickstart.

CHAPTER 4

Package Api Documentation for potranslator

4.1 API Reference for the classes in potranslator.potranslator.py

Main module.

class potranslator.potranslator.**PoTranslator**(*pot_dir=None*, *locale_dir=None*)
Bases: object

This is the main class of this library. This class manages all translation tasks.

Parameters

- **pot_dir** – string. Path to the pot directory.
- **locale_dir** – string. Path to the locale directory.

translate(*file_name*, *target_lang='auto'*, *src_lang='auto'*, *encoding='utf-8'*, *auto_save=False*, *compiled=False*)
Translates the given po file in the specified target language.

Parameters

- **file_name** – string. Path to the filename of the file to translate.
- **target_lang** – string. Target language for translation.
- **src_lang** – string. Source language for translation.
- **encoding** – string. Encoding for saving the po files.
- **auto_save** – bool. Toggles auto save feature.
- **compiled** – bool. Toggles compilation to mo files.

Returns tuple. A tuple containing the translated version of the original catalog and the status of the POFile.

translate_all_locale(*src_lang='auto'*, *encoding='utf-8'*, *auto_save=False*, *compiled=False*)
Translates all the po files in the found languages in the locale folder.

Parameters

- **src_lang** – string. Source language for translation.
- **encoding** – string. Encoding for saving the po files.
- **auto_save** – bool. Toggles auto save feature.
- **compiled** – bool. Toggles compilation to mo files.

Returns Dictionary. A dictionary of po files.

translate_from_pot(*filename*, *status*, *target_langs*, *src_lang='auto'*, *encoding='utf-8'*, *auto_save=False*, *compiled=False*)

Translates the given pot file in the specified target languages.

Parameters

- **filename** – string. Path to the filename of the file to translate.
- **target_langs** – sequence of strings. Target language for translation.
- **src_lang** – string. Source language for translation.
- **encoding** – string. Encoding for saving the po files.
- **auto_save** – bool. Toggles auto save feature.
- **compiled** – bool. Toggles compilation to mo files.

Returns Dictionary. A dictionary of po files.

translate_all_pot(*target_langs*, *src_lang='auto'*, *encoding='utf-8'*, *auto_save=False*, *compiled=False*)

Translates all the pot files in the pot folder in the specified target languages.

Parameters

- **target_langs** – sequence of strings. Target language for translation.
- **src_lang** – string. Source language for translation.
- **encoding** – string. Encoding for saving the po files.
- **auto_save** – bool. Toggles auto save feature.
- **compiled** – bool. Toggles compilation to mo files.

Returns Dictionary. A dictionary of po files.

CHAPTER 5

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:

5.1 Types of Contributions

5.1.1 Report Bugs

Report bugs at https://github.com/SekouD/potranslator/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.

5.1.2 Fix Bugs

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

5.1.3 Implement Features

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

5.1.4 Write Documentation

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

5.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/SekouD/potranslator/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 :)

5.2 Get Started!

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

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

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

5.3 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 2.7, 3.4, 3.5 and 3.6, and for PyPy. Check https://travis-ci.org/SekouD/potranslator/pull_requests and make sure that the tests pass for all supported Python versions.

5.4 Tips

To run a subset of tests:

```
$ py.test tests.test_potranslator
```

5.5 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:

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

Travis will then deploy to PyPI if tests pass.

CHAPTER 6

Credits

6.1 Development Lead

- SekouD <sekoud.python@gmail.com> GPG key ID: B51D1046EF63C50B

6.2 Contributors

None yet. Why not be the first?

CHAPTER 7

History

7.1 1.1.0 (2018-07-08)

- Now uses importlib_ressources for faster startup from CLI.
- Updated the command line usability.
- Added Type Annotation compliant with PEP 561.
- Updated Documentation.

7.2 1.0.5 (2018-07-06)

- Updated Documentation.
- Translated the documentation in French, Spanish, Italian, German, Italian, Japanese and Chinese.
- More detailed updates to the po files meta-data.

7.3 1.0.0 (2018-07-05)

- First release candidate.
- Added Command Line Interface.

7.4 0.1.0 (2018-06-27)

- First release on PyPI.

CHAPTER 8

Indices and tables

- genindex
- modindex
- search

Python Module Index

p

Index

P

T