# **Translate Toolkit Documentation**

Release 2.4.0

Translate.org.za

# Contents

1	User	's Guide	3
	1.1	Features	3
	1.2	Installation	4
	1.3	Converters	6
	1.4	Tools	61
	1.5	Scripts	100
	1.6	Use Cases	112
	1.7	Translation Related File Formats	129
2	Deve	eloper's Guide	163
	2.1	Translate Styleguide	163
	2.2	Documentation	170
	2.3	Building	173
	2.4	Testing	174
	2.5	Command Line Functional Testing	176
	2.6	Contributing	178
	2.7	Translate Toolkit Developers Guide	180
	2.8	Making a Translate Toolkit Release	
	2.9	Deprecation of Features	
3	Addi	itional Notes	191
	3.1	Release Notes	191
	3.2	History of the Translate Toolkit	229
	3.3	License	230
4	API	Reference	231
	4.1	API	231
Рy	thon I	Module Index	699
In	dex		703

 $We lcome \ to \ Translate \ Toolkit's \ documentation. \ This \ documentation \ covers \ both \ user's \ and \ programmer's \ perspective.$ 

Contents 1

2 Contents

# CHAPTER 1

User's Guide

This part has the user's documentation for the tools included in the Translate Toolkit.

# 1.1 Features

- Work with **ONE localisation format**. You'll no longer be editing DTD files in one tool, .properties in another, OpenOffice GSI in a third. Simply do all your localisation in a PO or XLIFF editor
- Converters for a number of *formats* 
  - OpenOffice.org SDF/GSI
  - Mozilla: .properties, DTD, XHTML, .inc, .ini, etc
  - Others: Comma Separated Value, TMX, XLIFF, TBX, PHP, WordFast TXT, Qt .ts, txt, .ini, Windows .rc, ical, subtitles, Mac OS X strings
- File access to localization files through the format API in all the above formats, as well as .qph, .qm, .mo
- Output valid target file types. We make sure that your output files (e.g. .properties) contain all comments from the original file and preserves the layout of the original as far as possible. If your PO entry is marked as fuzzy we use the English text, not your half complete translation. The converters for OpenOffice.org and Mozilla formats will also perform simple checks and corrections to make sure you have none of those hard to find localisation bugs.
- Our checker has over 42 checks to find errors such as: missing or translated variables, missing accelerator keys, bad escaping, start capitalisation, missing sentences, bad XML and much more.
- · Language awareness, taking language conventions for capitalisation, quotes and other punctuation into account
- Find conflicting translations easily, cases where you have translated a source word differently or used a target word for 2 very different English concepts
- Extract messages using simple text or a regular expression allowing you to quickly find and extract words that you need to fix due to glossary changes.
- Merge snippets of PO files into your existing translations.

- Create word, string and file **counts** of your files. Making it much easier to budget time as string counts do not give you a good indication of expected work.
- Create a set of PO files with **debugging** entries to allow you to easily locate the source of translations. Very useful in OpenOffice.org which provides scant clues as to where the running application has sourced the message.

The Translate Toolkit is also a **powerful API** for writing translation and localisation tools, already used by our own and several other projects. See the *base class* section for more information.

# 1.2 Installation

This is a guide to installing the Translate Toolkit on your system. If the Translate Toolkit is already packaged for your system, this is probably the easiest way to install it. For several Linux distributions, the package might be available through your package manager. On Windows, we recommend using a virtual environment.

If your system already has the toolkit prepackaged, then please let us know what steps are required to install it.

# 1.2.1 Building

For build instructions, see the *Building* page.

#### 1.2.2 Download

Download a stable released version. Or if you have a python environment, run *pip install translate-toolkit*. For those who need problems fixed, or who want to work on the bleeding edge, get the latest source from *Git*.

If you install through your distribution's package manager, you should automatically have all the dependencies you need. If you are installing a version from Version Control, or from a source release, you should check the README file for information on the dependencies that are needed. Some of the dependencies are optional. The README file documents this.

# 1.2.3 Installing packaged versions

Get the package for your system:

RPM	If you want to install easily on an RPM based system
.tar.gz	for source based installing on Linux
.deb	for Debian GNU/Linux (etch version)

The RPM package can be installed by using the following command:

```
$ rpm -Uvh translate-toolkit-1.0.1.rpm
```

To install a tar.bz2:

```
$ tar xvjf translate-toolkit-1.1.0.tar.bz2
$ cd translate-toolkit-1.1.0
$ su
$ ./setup.py install
```

On Debian (if you are on etch), just type the following command:

```
$ aptitude install translate-toolkit
```

If you are using an old Debian stable system, you might want to install the .tar.bz2 version. Be sure to install python and python development first with:

```
$ apt-get install python python-dev
```

Alternatively newer packages might be in testing.

# 1.2.4 Installing on Windows

On Windows we recommend that you install Translate Toolkit using a virtual environment. This makes installation clean and isolated.

Use the latest Python 2.7 (at least Python 2.7.9 or newer as it bundles the pip installer). Install virtualenvwrapper-win to simplify handling of virtualenvs.

- 1. Install latest Python 2.7
- 2. Open cmd.exe or similar
- 3. pip install virtualenvwrapper-win
- 4. mkvirtualenv ttk where "ttk" is the name for the new virtualenv
- 5. pip install translate-toolkit to install latest stable or pip install -pre translate-toolkit to try a pre-release
- 6. pip install lxml>=3.5 to be able to use XLIFF or other XML formats
- 7. po2prop -version to double check you have the right version

Next times you need to use Translate Toolkit just remember to:

- 1. Open cmd.exe or similar
- 2. workon ttk to enable the virtualenv again
- 3. Run the Translate Toolkit commands you want

# 1.2.5 Installing from Git

If you want to try the bleeding edge, or just want to have the latest fixes from a stabilising branch then you need to use Git to get your sources:

```
$ git clone https://github.com/translate/translate.git
```

This will retrieve the master branch of the Toolkit. Further Git instructions are also available.

Once you have the sources you have two options, a full install:

```
$ su
$ ./setup.py install
```

or, running the tools from the source directory:

```
$ su
$ pip install -e .
```

1.2. Installation 5

# 1.2.6 Verify installed version

To verify which version of the toolkit you have installed run:

```
$ prop2po --version
prop2po 2.4.0
```

# 1.2.7 Cleaning up existing installation

To remove old versions of the toolkit which you might have installed without a virtual environment or without your package manager.

The following advice only applies to manual installation from a tarball.

1. Find location of your python packages:

2. Delete toolkit package from your Python site-packages directory e.g.:

```
$ rm -R /usr/local/lib/python2.7/dist-packages/translate
```

# 1.3 Converters

# 1.3.1 General Usage

The tools follow a general usage convention which is helpful to understand.

### **Input & Output**

The last two arguments of your command are the input and output files/directories:

```
moz2po <input> <output>
```

You can of course still use the -i and -o options which allows you to reorder commands

```
moz2po -o <output> -i <input>
```

### **Error Reporting**

All tools accept the option --errorlevel. If you find a bug, add this option and send the traceback to the developers.

```
moz2po <other-options> --errorlevel=traceback
```

# **Templates**

If you are working with any file format and you wish to preserve comments and layout then use your source file as a template.

```
po2dtd -t <source-file> <input> <output>
```

This will use the files in <source-file> as a template, merge the PO files in <input>, and create new DTD files in <output>

If you ran this without the templates you would get valid DTD files but they would not preserve the layout or all the comments from the source DTD file

The same concept of templates is also used when you merge files.

```
pomerge -t <old> <fixes> <new>
```

This would take the <old> files merge in the <fixes> and output new PO files, preserving formatting, into <new>.

You can use the same directory for <old> and <new> if you want the merges to overwrite files in <old>.

# source2target

The converters all follow this convention:

- source = the format from which you are converting e.g. in oo2po we are converting from OpenOffice.org SDF/GSI
- target = the format into which you are converting e.g. in 002po we are converting to Gettext PO

# **Getting Help**

The --help option will always list the available commands for the tool.

```
moz2po --help
```

# 1.3.2 moz2po

moz2po converts Mozilla files to PO files. It wraps converters that handle .properties, .dtd and some strange Mozilla files. The tool can work with files from Mozilla's Mercurial repository. The tools thus provides a complete roundtrip for Mozilla localisation using PO files and PO editors.

**Note:** This page should only be used as a reference to the command-line options for moz2po and po2moz. For more about using the Translate Toolkit and PO files for translating Mozilla products, please see the page on *Mozilla L10n Scripts*.

# **Usage**

```
moz2po [options] <dir> <po>po2moz [options] <po> <dir>
```

Where:

<dir></dir>	is a directory containing valid Mozilla files
<po></po>	is a directory of PO or POT files

Options (moz2po):

--version show program's version number and exit -h, --help show this help message and exit output a manpage based on the help --manpage --progress=PROGRESS show progress as: dots, none, bar, names, verbose --errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback -i INPUT, --input=INPUT read from INPUT in inc, it, \*, dtd, properties formats -x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths -o OUTPUT, --output=OUTPUT write to OUTPUT in it.po, it.pot, manifest, xhtml.po, xhtml.pot, ini.po, ini.pot, rdf, js, \*, html.po, html.pot, inc.po, inc.pot, dtd.po, dtd.pot, properties.po, properties.pot formats -t TEMPLATE, --template=TEMPLATE read from TEMPLATE in it, \*, properties, dtd, inc formats -S, --timestamp skip conversion if the output file has newer timestamp -P, --pot output PO Templates (.pot) rather than PO files (.po) --duplicates=DUPLICATESTYLE what to do with duplicate strings (identical source text): merge, msgctxt (default: 'msgctxt') Options (po2moz): --version show program's version number and exit -h, --help show this help message and exit --manpage output a manpage based on the help **--progress=PROGRESS** show progress as: dots, none, bar, names, verbose --errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback -i INPUT, --input=INPUT read from INPUT in dtd.po, dtd.pot, ini.po, ini.pot, inc.po, inc.pot, manifest, it.po, it.pot, \*, html.po, html.pot, js, rdf, properties.po, properties.pot, xhtml.po, xhtml.pot formats -x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths -o OUTPUT, --output=OUTPUT write to OUTPUT in dtd, \*, inc, it, properties formats -t TEMPLATE, --template=TEMPLATE read from TEMPLATE in dtd, \*, inc, it, properties formats -S, --timestamp skip conversion if the output file has newer timestamp -l LOCALE, --locale=LOCALE set output locale (required as this sets the directory names) **--removeuntranslated** remove untranslated strings from output **--threshold=PERCENT** only convert files where the translation completion is above PERCENT --fuzzy use translations marked fuzzy

don't use translations marked fuzzy (default)

# **Examples**

# **Creating POT files**

--nofuzzy

See also:

#### Creating Mozilla POT files.

After extracting the en-US 110n files, you can run the following command:

```
moz2po -P 110n/en-US pot
```

This creates a set of POT (-P) files in the pot directory from the Mozilla files in 110n/en-US for use as PO Templates.

If you want to create a set of POT files with another base language try the following:

```
moz2po -P 110n/fr-FR fr-pot
```

This will create a set of POT files in fr-pot that have French as your source language.

# Creating PO files from existing non-PO translations

If you have existing translations (Mozilla related or other Babelzilla files) and you wish to convert them to PO for future translation then the following generic instructions will work:

```
moz2po -t en-US af-ZA af-ZA_pofiles
```

This will combine the untranslated template en-US files from en-US combine them with your existing translations in af-ZA and output PO files to af-ZA\_pofiles.

```
moz2po -t l10n/fr l10n/xh po/xh
```

For those who are not English fluent you can do the same with another languages. In this case msgid will contain the French text from 110n/fr. This is useful for translating where the translators other languages is not English but French, Spanish or Portuguese. Please make sure that the source languages i.e. the msgid language is fully translated as against en-US.

### **Creating Mercurial ready translations**

```
po2moz -t 110n/en-US po/xh 110n/xh
```

Create Mozilla files using the templates files in 110n/en-US (see above for how to create them) with PO translations in po/xh and output them to 110n/xh. The files now in 110n/xh are ready for submission to Mozilla and can be used to build a language pack or translated version of Mozilla.

#### Issues

You can perform the bulk of your work (99%) with moz2po.

Localisation of XHTML is not yet perfect, you might want to work with the files directly.

Issue 203 tracks the outstanding features which would allow complete localisation of Mozilla including; all help, start pages, rdf files, etc. It also tracks some bugs.

Accesskeys don't yet work in .properties files and in several cases where the Mozilla .dtd files don't follow the normal conventions, for example in security/manager/chrome/pippki/pref-ssl.dtd.po. You might also want to check the files mentioned in this Mozilla bug 329444 where mistakes in the DTD-definitions cause problems in the matching of accelerators with the text.

You might want to give special attention to the following files since it contains customisations that are not really translations.

- mail/chrome/messenger/downloadheaders.dtd.po
- · toolkit/chrome/global/intl.properties.po

Also, all width, height and size specifications need to be edited with feedback from testing the translated interfaces.

There are some constructed strings in the Mozilla code which we can't do much about. Take good care to read the localisation notes. For an example, see mail/chrome/messenger/downloadheaders.dtd.po. In that specific file, the localisation note from the DTD file is lost, so take good care of those.

The file extension of the original Mozilla file is required to tell the Toolkit how to do the conversion. Therefore, a file like foo.dtd must be named foo.dtd.po in order to *po2moz* to recognise it as a DTD file.

# 1.3.3 oo2po

Convert between OpenOffice.org GSI/SDF files and the PO format. This tool provides a complete roundtrip; it preserves the structure of the GSI file and creates completely valid PO files.

oo2xliff will convert the SDF files to XLIFF format.

### Usage

```
oo2po [options] <sdf> <output>
po2oo [options] [-t <en-US.sdf>] -l <targetlang> <input> <sdf|output>
```

### or for XLIFF files:

```
oo2xliff [options] -1 <targetlang> <sdf> <output>
xliff2oo [options] [-t <en-US.sdf>] -1 <targetlang> <input> <sdf|output>
```

### Where:

<sdf></sdf>	is a valid OpenOffice.org GSI or SDF files
<output></output>	is a directory for the resultant PO/POT/XLIFF files
<input/>	is a directory of translated PO/XLIFF files
<targetlang></targetlang>	is the ISO 639 language code used in the sdf file, e.g. af

Options (oo2po and oo2xliff):

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

--progress=PROGRESS show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in oo, sdf formats

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in po, pot, xlf formats

**-S, --timestamp** skip conversion if the output file has newer timestamp

**-P, --pot** output PO Templates (.pot) rather than PO files (.po) (only available in oo2po

```
oo2xliff)
      --source-language=LANG set source language code (default en-US)
      --nonrecursiveinput don't treat the input oo as a recursive store
      --duplicates=DUPLICATESTYLE what to do with duplicate strings (identical source text): merge,
                           msgctxt (default: 'msgctxt')
      --multifile=MULTIFILESTYLE how to split po/pot files (single, toplevel or onefile)
Options (po2oo and xliff2oo):
      --version
                           show program's version number and exit
      -h, --help
                           show this help message and exit
                           output a manpage based on the help
      --manpage
      --progress=PROGRESS show progress as: dots, none, bar, names, verbose
      --errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback
      -i INPUT, --input=INPUT read from INPUT in po, pot, xlf formats
      -x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths
      -o OUTPUT, --output=OUTPUT write to OUTPUT in oo, sdf formats
      -t TEMPLATE, --template=TEMPLATE read from TEMPLATE in oo, sdf formats
                           skip conversion if the output file has newer timestamp
      -S. --timestamp
      -l LANG, --language=LANG set target language code (e.g. af-ZA) [required]
      --source-language=LANG set source language code (default en-US)
      -T, --keeptimestamp don't change the timestamps of the strings
      --nonrecursiveoutput don't treat the output oo as a recursive store
      --nonrecursivetemplate don't treat the template oo as a recursive store
      --skipsource
                           don't output the source language, but fallback to it where needed
      --filteraction=ACTION action on pofilter failure: none (default), warn, exclude-serious, exclude-all
      --threshold=PERCENT only convert files where the translation completion is above PERCENT
      --fuzzy
                           use translations marked fuzzy
      --nofuzzy
                           don't use translations marked fuzzy (default)
      --multifile=MULTIFILESTYLE how to split po/pot files (single, toplevel or one file)
```

-l LANG, --language=LANG set target language to extract from oo file (e.g., af-ZA) (required for

#### **Examples**

These examples demonstrate most of the useful invocations of oo2po:

#### **Creating POT files**

```
oo2po -P en-US.sdf pot
```

Extract messages from en-US.sdf and place them in a directory called pot. The  $\neg P$  option ensures that we create POT files instead of PO files.

```
oo2po -P --source-language=fr fr-FR.sdf french-pot
```

Instead of creating English POT files we are now creating POT files that contain French in the msgid. This is useful for translators who are not English literate. You will need to have a fully translated sdf in the source language.

# Creating PO files from existing work

```
oo2po --duplicates=merge -l zu zu-ZA.sdf zulu
```

Extract all existing Zulu (*zu*) messages from *zu-ZA.sdf* and place them in a directory called *zulu*. If you find duplicate messages in a file then merge them into a single message (This is the default behaviour for traditional PO files). You might want to use *pomigrate2* to ensure that your PO files match the latest POT files.:

```
cat GSI_af.sdf GSI_xh.sdf > GSI_af-xh.sdf
oo2po --source-language=af -l xh GSI_af-xh.sdf af-xh-po
```

Here we are creating PO files with your existing translations but a different source language. Firstly we combine the two SDF files. Then oo2po creates a set of PO files in *af-xh-po* using Afrikaans (*af*) as the source language and Xhosa (*xh*) as the target language from the combined SDF file *GSI\_af-xh.sdf* 

# Creating a new GSI/SDF file

```
po2oo -l zu zulu zu_ZA.sdf
```

Using PO files found in *zulu* create an SDF files called *zu\_ZA.sdf* for language *zu*:

```
po2oo -l af -t en-US.sdf --nofuzzy --keeptimestamp --filteraction=exclude-serious_

→afrikaans af_ZA.sdf
```

Create an Afrikaans (af) SDF file called af\_ZA.sdf using en-US.sdf as a template and preserving the timestamps within the SDF file while also eliminating any serious errors in translation. Using templates ensures that the resultant SDF file has exactly the same format as the template SDF file. In an SDF file each translated string can have a timestamp attached. This creates a large amount of unuseful traffic when comparing version of the SDF file, by preserving the timestamp we ensure that this does not change and can therefore see the translation changes clearly. We have included the nofuzzy option (on by default) that prevent fuzzy PO messages from getting into the SDF file. Lastly the filteraction option is set to exclude serious errors: variables failures and translated XML will be excluded from the final SDF.

### helpcontent2

The escaping of helpcontent2 from SDF files was very confusing, issue 295 implemented a fix that appeared in version 1.1.0 (All known issues were fixed in 1.1.1). Translators are now able to translate helpcontent2 with clean escaping.

# 1.3.4 odf2xliff and xliff2odf

Convert OpenDocument (ODF) files to XLIFF localization files. Create translated ODF files by combining the original ODF files with XLIFF files containing translations of strings in the original document.

XLIFF is the XML Localization Interchange File Format developed by OASIS (The Organization for the Advancement of Structured Information Standards) to allow translation work to be standardised no matter what the source format and to allow the work to be freely moved from tool to tool.

If you are more used to software translation or 110n, you might want to read a bit about *Document translation*. This should help you to get the most out of translating ODF with XLIFF.

# **Usage**

```
odf2xliff [options] <original_odf> <xliff> xliff2odf [options] -t <original_odf> <xliff> <translated_odf>
```

#### Where:

<original_odf></original_odf>	is an ODF document whose strings have to be translated
<xliff></xliff>	is an XLIFF file
<translated_odf></translated_odf>	is an ODF file to generate by replacing the strings in
	<pre><original_odf> with the translated strings in <xliff></xliff></original_odf></pre>

# Options (odf2xliff):

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

--progress=PROGRESS show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in ODF format

-o OUTPUT, --output=OUTPUT write to OUTPUT in XLIFF format

**-S, --timestamp** skip conversion if the output file has newer timestamp

# Options (xliff2odf):

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

**--progress=PROGRESS** show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in XLIFF formats

-o OUTPUT, --output=OUTPUT write to OUTPUT in ODF format

-t TEMPLATE, --template=TEMPLATE read from TEMPLATE in ODF format

-S, --timestamp skip conversion if the output file has newer timestamp

### **Examples**

```
odf2xliff english.odt english_français.xlf
```

Create an XLIFF file from an ODT file (the source ODF file could also be any of the other ODF files, including ODS, ODG, etc.).

```
xliff2odf -t english.odt english_français.xlf français.odt
```

Using english.odt as the template document, and english\_français.xlf as the file of translations, create a translated file français.odt.

# **Bugs**

This filter is not yet extensively used – we appreciate your feedback. For more information on conformance to standards, see the *XLIFF* or *OpenDocument Format* pages.

# 1.3.5 prop2po

Convert between Java property files (.properties) and Gettext PO format.

Note: this tool completely eliminates the need for *native2ascii* as po2prop does the correct escaping to the Latin1 encoding that is needed by Java.

The following other formats are also supported via the *-personality* parameter:

- · Adobe Flex
- · Skype .lang
- · Mac OS X .strings
- Mozilla .properties

#### **Usage**

#### Where:

<pre><pre><pre><pre>property&gt;</pre></pre></pre></pre>	is a directory containing property files or an individual property file	
<po></po>	is a directory containing PO files and an individual property file	
<template></template>	is a directory of template property files or a single template property file	

### Options (prop2po):

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

--progress=PROGRESS show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in properties format

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in po, pot formats

```
-t TEMPLATE, --template=TEMPLATE read from TEMPLATE in properties format
```

**-S, --timestamp** skip conversion if the output file has newer timestamp

**-P, --pot** output PO Templates (.pot) rather than PO files (.po)

--personality=TYPE override the input file format: flex, java, mozilla, java-utf8, skype, gaia, strings

(for .properties files, default: java)

--encoding=ENCODING override the encoding set by the personality

--duplicates=DUPLICATESTYLE what to do with duplicate strings (identical source text): *merge*, *msgctxt* (default: 'msgctxt')

# Options (po2prop):

**--version** show program's version number and exit

**-h, --help** show this help message and exit

**--manpage** output a manpage based on the help

--progress=PROGRESS show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in po, pot formats

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in properties format

-t TEMPLATE, --template=TEMPLATE read from TEMPLATE in properties format

**-S, --timestamp** skip conversion if the output file has newer timestamp

--personality=TYPE override the input file format: flex, java, mozilla, java-utf8, skype, gaia, strings (for .properties files, default: java)

**--encoding=ENCODING** override the encoding set by the personality (since 1.8.0)

--removeuntranslated remove untranslated strings from output

--threshold=PERCENT only convert files where the translation completion is above PERCENT

**--fuzzy** use translations marked fuzzy

**--nofuzzy** don't use translations marked fuzzy (default)

# **Examples**

These examples demonstrate most of the useful invocations of prop2po:

### **Creating POT files**

```
prop2po -P properties pot
```

Extract messages from *properties* directory and place them in a directory called *pot*. The  $\neg P$  option ensures that we create POT files instead of PO files.:

```
prop2po -P file.properties file.pot
```

Extract messages from file.properties and place them in file.pot.

# Creating PO files from existing work

```
prop2po --duplicates=msgctxt -t reference zu zu-po
```

Extract all existing Zulu messages from *zu* directory and place the resultant PO files in a directory called *zu-po*. If you find duplicate messages in a file then use Gettext's mgsctxt to disambiguate them. During the merge we use the properties files in *reference* as templates and as the source of the English text for the msgid. Once you have your PO files you might want to use *pomigrate2* to ensure that your PO files match the latest POT files.

# Creating .properties files from your translations

```
po2prop -t reference zu-po zu
```

Using our translations found in zu-po and the templates found in reference we create a new set of property files in zu. These new property files will look exactly like those found in the templates, but with the text changed to the translation. Any fuzzy entry in our PO files will be ignored and any untranslated item will be placed in zu in English. The properties file created will be based on the Java specification and will thus use escaped Unicode. Where:

#### Will appear in the files as:

```
\u1E7D\u1E01\u1E3D\u1E7B\u1E1D
```

To get output as used by Mozilla localisation do the following:

```
po2prop --personality=mozilla -t reference zu-po zu
```

This will do exactly the same as above except that the output will now appear as real Unicode characters in UTF-8 encoding.

#### Doing away with native2ascii

The native2ascii command is the traditional tool of property file localisers. With prop2po there is no need to use this command or to ever work directly with the escaped Unicode.

If you are working mostly with Gettext PO files then this is a double benefit as you can now use your favourite PO editor to translate Java applications. Your process would now look like this:

```
prop2po some.properties some.po
```

Firstly create a PO file that you can translate. Now translate it in your favourite PO editor.:

```
po2prop -t some.properties some.po some-other.properties
```

Using the original properties file as a template we preserve all layout and comments, combined with your PO translation we create a new translate properties file. During this whole process we have not needed to understand or process any escaping prop2po and po2prop handle that all automatically.

If you have existing translations you can recover them as follows:

```
prop2po -t some.properties translations.properties translations.po
```

This takes the default English properties file and combines it with your translate properties file and created a PO file. You now continue translating using your PO file.

# 1.3.6 php2po

Converts PHP localisable string arrays to Gettext PO format.

# **Usage**

```
php2po [options] <php> <po>
po2php [options] <po> <php>
```

# Where:

<php></php>	is a valid PHP localisable file or directory of those files
<po></po>	is a directory of PO or POT files

# Options (php2po):

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

--progress=PROGRESS show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in php format

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in po, pot formats

-t TEMPLATE, --template=TEMPLATE read from TEMPLATE in php format

-S, --timestamp skip conversion if the output file has newer timestamp
-P, --pot output PO Templates (.pot) rather than PO files (.po)

--duplicates=DUPLICATESTYLE what to do with duplicate strings (identical source text): merge,
msgctxt (default: 'msgctxt')

# Options (po2php):

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

--progress=PROGRESS show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in po, pot formats

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in php format

-t TEMPLATE, --template=TEMPLATE read from TEMPLATE in php format

**-S, --timestamp** skip conversion if the output file has newer timestamp

--threshold=PERCENT only convert files where the translation completion is above PERCENT

**--fuzzy** use translations marked fuzzy

**--nofuzzy** don't use translations marked fuzzy (default)

# **Formats Supported**

Check *PHP format* document to see to which extent the PHP format is supported.

# **Examples**

This example looks at roundtrip of PHP translations as well as recovery of existing translations.

First we need to create a set of POT files.:

```
php2po -P lang/en pot/
```

All .php files found in the lang/en directory are converted to Gettext POT files and placed in the pot directory.

If you are translating for the first time then you can skip the next step. If you need to recover your existing translations then we do the following:

```
php2po -t lang/en lang/zu po-zu/
```

Using the English PHP files found in lang/en and your existing Zulu translation in lang/zu we create a set of PO files in po-zu. These will now have your translations. Please be aware that in order for that to work 100% you need to have both English and Zulu at the same revision, if they are not you will have to review all translations.

You are now in a position to translate your recovered translations or your new POT files.

Once translated you can convert back as follows:

```
po2php -t lang/en po-zu/ lang/zu
```

Your translations found in the Zulu PO directory, po-zu, will be converted to PHP using the files in lang/en as templates and placing your new translations in lang/zu.

To update your translations simply redo the POT creation step and make use of *pot2po* to bring your translation up-to-date.

# 1.3.7 sub2po

sub2po allows you to use the same principles of PO files with *Subtitles*. In PO only items that change are marked fuzzy and only new items need to be translated, unchanged items remain unchanged for the translation.

# **Usage**

```
sub2po [options] <foo.srt> <foo.po>
po2sub [options] [-t <foo.srt>] <XX.po> <foo-XX.srt>
```

# Where:

foo.srt	is the input subtitle file
foo.po	is an empty PO file that may be translated
XX.po	is a PO file translated into the XX language
foo-XX.srt	is the foo.srt file translated into language XX

# Options (sub2po):

```
--version show program's version number and exit
```

-h, --help show this help message and exit--manpage output a manpage based on the help

--progress=PROGRESS show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in .srt format

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in po, pot formats

-t TEMPLATE, --template=TEMPLATE read from TEMPLATE in ass, srt, ssa, sub formats

-S, --timestamp skip conversion if the output file has newer timestamp
-P, --pot output PO Templates (.pot) rather than PO files (.po)

--duplicates=DUPLICATESTYLE what to do with duplicate strings (identical source text): merge,
msgctxt (default: 'msgctxt')

# Options (po2sub):

**--version** show program's version number and exit

**-h, --help** show this help message and exit

**--manpage** output a manpage based on the help

**--progress=PROGRESS** show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in po, pot formats

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in srt format

-t TEMPLATE, --template=TEMPLATE read from TEMPLATE in txt format

**-S, --timestamp** skip conversion if the output file has newer timestamp

**--threshold=PERCENT** only convert files where the translation completion is above PERCENT

**--fuzzy** use translations marked fuzzy

**--nofuzzy** don't use translations marked fuzzy (default)

### **Examples**

To create the POT files is simple:

```
sub2po -P SUBTITLE_FILE subtitles.pot
```

A translator would copy the POT file to their own PO file and then create translations of the entries. If you wish to create a PO file and not a POT file then leave off the -P option.

To convert back:

```
po2sub -t SUBTITLE_FILE subtitles-XX.po subtitles-XX.srt
```

# **Translating**

Translate as normal. However, see the issues mentioned at *Subtitles*.

# **Bugs**

There might be some issues with encodings, since the srt files don't specify them. We assume files to be encoded in UTF-8, so a conversion should solve this easily. Note that most of the handling of the srt files come from gaupol.

# 1.3.8 txt2po

txt2po allows you to use the same principles of PO files with normal text files. In PO only items that change are marked fuzzy and only new items need to be translated, unchanged items remain unchanged for the translation.

# **Usage**

```
txt2po [options] <foo.txt> <foo.po>
po2txt [options] [-t <foo.txt>] <XX.po> <foo-XX.txt>
```

#### Where:

foo.txt	is the input plain text file	
foo.po	is an empty PO file that may be translated	
XX.po	is a PO file translated into the XX language	
foo-XX.txt	is the foo.txt file translated into language XX	

### Options (txt2po):

**--version** show program's version number and exit

**-h, --help** show this help message and exit

**--manpage** output a manpage based on the help

--progress=PROGRESS show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in \*, txt formats

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in po, pot formats

**-S**, **--timestamp** skip conversion if the output file has newer timestamp

**-P, --pot** output PO Templates (.pot) rather than PO files (.po)

**--encoding=ENCODING** The encoding of the input file (default: UTF-8)

--flavour=FLAVOUR The flavour of text file: plain (default), dokuwiki, mediawiki

**--no-segmentation** Don't segment the file, treat it like a single message

--duplicates=DUPLICATESTYLE what to do with duplicate strings (identical source text): *merge*, *msgctxt* (default: 'msgctxt')

# Options (po2txt):

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

--progress=PROGRESS show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in po, pot formats

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in txt format

-t TEMPLATE, --template=TEMPLATE read from TEMPLATE in txt format

**-S, --timestamp** skip conversion if the output file has newer timestamp

**--encoding=ENCODING** The encoding of the template file (default: UTF-8)

-w WRAP, --wrap=WRAP set number of columns to wrap text at

--threshold=PERCENT only convert files where the translation completion is above PERCENT

**--fuzzy** use translations marked fuzzy

**--nofuzzy** don't use translations marked fuzzy (default)

#### A roundtrip example

# **Preparing input files**

With **txt2po** a text file is broken down into sections. Each section is separated by a line of whitespace. Each section will appear as a msgid in the PO file. Because of this simple method of breaking up the input file it might be necessary to alter the layout of your input file. For instance you might want to separate a heading from a paragraph by using whitespace.

For steps in a process you would want to leave a blank line between each step so that each step can be translated independently.

For a list of items you might want to group them together so that a translator could for example place them in alphabetic order for their translation.

Once the input file is prepared you can proceed to the next step.

### Creating the POT files

#### This is simple:

```
txt2po -P TEXT_FILE text_file.pot
```

A translator would copy the POT file to their own PO file and then create translations of the entries. If you wish to create a PO file and not a POT file then leave off the -P option.

You might want to manually edit the POT file to remove items that should not be translated. For instance if part of the document is a license you might want to remove those if you do not want the license translated for legal reasons.

# **Translating**

Translate as normal. However translators should be aware that writers of the text file may have used spaces, dashes, equals, underscores and other aids to indicate things such as:

```
* Headings and sub-headings

* Code examples, command lines examples

* Various lists

* etc
```

They will need to adapt these to work in their language being aware of how they will appear once they are merged with the original text document.

# Creating a translated text file

With the translations complete you can create a translated text file like this:

```
po2txt -w 75 -t TEXT_FILE translated.po TEXT_FILE.translated
```

This uses the original text file as a template and creates a new translated text file using the translations found in the PO file

The -w command allows you to reflow the translated text to N number of characters, otherwise the text will appear as one long line.

# Help with Wiki syntax

#### dokuwiki

To retrieve the raw syntax for your dokuwiki page add '?do=export\_raw' to you URL. The following would retrieve the DokuWiki home page in raw dokuwiki format https://www.dokuwiki.org/dokuwiki?do=export\_raw

```
wget https://www.dokuwiki.org/dokuwiki?do=export_raw -0 txt2po.txt
txt2po --flavour=dokuwiki -P txt2po.txt txt2po.pot
# edit txt2po.pot
po2txt -t txt2po.txt fr.po fr.txt
```

First we retrieve the file in raw dokuwiki format, then we create a POT file for editing. We created a French translation and using po2txt plus the original file as a template we output fr.txt which is a French version of the original txt2po.txt. This file can now be uploaded to the wiki server.

# MediaWiki

To retrieve the raw media wiki syntax add '?action=raw' to you wiki URL. The following retrieves the Translate Toolkit page from Wikipedia in raw MediaWiki format http://en.wikipedia.org/wiki/Translate\_Toolkit?action=raw or http://en.wikipedia.org/w/index.php?title=Pootle&action=raw.

To process follow the instructions above but substituting the MediaWiki retrieval method.

# 1.3.9 po2wordfast

Convert Gettext PO files to a Wordfast Translation Memory translation memory file.

Wordfast is a popular Windows based computer-assisted translation tool.

#### **Usage**

```
po2wordfast [options] --language <target> <po> <wordfast>
```

#### Where:

<po></po>	a PO file or directory
<wordfast></wordfast>	a Wordfast translation memory file

### Options:

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

**--progress=PROGRESS** show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in po, pot formats

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in tmx format

**-S, --timestamp** skip conversion if the output file has newer timestamp

-l LANG, --language=LANG set target language code (e.g. af-ZA) [required]

**--source-language=LANG** set source language code (default: en)

# **Examples**

```
po2wordfast -l xh-ZA browser.po browser.txt
```

Use the Xhosa (xh-ZA) translations in the PO file browser.po to create a Wordfast translation memory file called browser.txt

# 1.3.10 po2tmx

Convert *Gettext PO* files to a *TMX* translation memory file. TMX is the Translation Memory eXchange format developed by OSCAR.

If you are interested in po2tmx, you might also be interested in *posegment* that can be used to perform some automated segmentation on sentence level.

### Usage

```
po2tmx [options] --language <target> <po> <tmx>
```

#### Where:

<po></po>	is a PO file
<tmx></tmx>	is a TMX file

### Options:

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

--progress=PROGRESS show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in po, pot formats

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in tmx format

**-S, --timestamp** skip conversion if the output file has newer timestamp

-l LANG, --language=LANG set target language code (e.g. af-ZA) [required]

**--source-language=LANG** set source language code (default: en)

**--comments=COMMENT** set default comment import: none, source, type or others (default: none)

### **Examples**

```
po2tmx -1 xh browser.po browser.tmx
```

Use the Xhosa (xh) translations in the PO file browser.po to create a TMX file called browser.tmx

#### **Bugs and issues**

### Markup stripping

po2tmx conforms to TMX v1.4 without stripping markup. See the *TMX* conformance page for more details. It has not been widely tested so your mileage may vary.

### TMX and PO in OmegaT

In some tools, like OmegaT, PO files are parsed without expanding escaped sequences, even though such tools use TMX for translation memory. Keep this in mind when using po2tmx, because po2tmx converts \n and \t to newlines and tabs in the TMX file. If such a TMX file is used while translating PO files in OmegaT, matching will be less than 100%.

In other tools, such as Swordfish, the PO comment "no-wrap" is interpreted in the same way as the equivalent function in XML, which may also lead to mismatches if TMXes from po2tmx are used.

There is nothing wrong with po2tmx, but if used in conjunction with tools that handle PO files differently, it may lead to less than perfect matching.

# **Tips**

# TMX with only unique segments

To create a TMX with no duplicates (in other words, only unique strings), use msgcat to first create a large PO file with non-uniques removed.

# 1.3.11 pot2po

Convert a Gettext PO Template file to a PO file and merge in existing translations if they are present. A translation memory (compendium) can also be used for fuzzy matching. This corresponds to a large extent with the program "msgmerge" from the gettext package.

Note: This tool also works with translation formats other than Gettext PO, for example XLIFF.

# **Usage**

```
pot2po [options] <pot> <po>
```

# Where:

_	is a PO Template (POT) file or directory of POT files
<po></po>	is a PO file or a directory of PO files

# Options:

--tm=TM

--version show program's version number and exit -h, --help show this help message and exit output a manpage based on the help **--progress=PROGRESS** show progress as: dots, none, bar, names, verbose --errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback -i INPUT, --input=INPUT read from INPUT in catkeys, lang, pot, ts, xlf, xliff formats -x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths -o OUTPUT, --output=OUTPUT write to OUTPUT in catkeys, lang, po, pot, ts, xlf, xliff formats -t TEMPLATE, --template=TEMPLATE read from TEMPLATE in catkeys, lang, po, pot, ts, xlf, xliff formats (old translations) -S, --timestamp skip conversion if the output file has newer timestamp -P, --pot output PO Templates (.pot) rather than PO files (.po)

1.3. Converters 25

The file to use as translation memory when fuzzy matching

- -s MIN\_SIMILARITY, --similarity=MIN\_SIMILARITY The minimum similarity for inclusion (default: 75%)
- --nofuzzymatching Disable all fuzzy matching

#### **Examples**

```
pot2po -t zu-1.0.1 pot-2.0.2 zu-2.0.2
```

Here we are initialising the PO files in zu-2.0.2 based on the POT files in pot-2.0.2. We are using the old translations in zu-1.0.1 as templates so that we can reuse our existing translations in the new files.

pot2po can also be used to update against newer templates an existing translation file in a format different than Gettext PO, for example XLIFF:

```
pot2po -t af.xlf -i templates.xlf -o updated-af.xlf
```

If the POT files have undergone major reshuffling then you may want to use *pomigrate2* which can now use pot2po as its merging backend. pomigrate2 will do its best to migrate your files to the correct locations before merging. It will also make use of a compendium if requested.

```
pot2po --tm=compendium.po --similarity=60 -t xh-old pot xh-new
```

With this update we are using *compendium.po* as a translations memory (you can make use of other files such as TMX, etc). We will accept any match that scores above 60%.

# Merging

It helps to understand when and how pot2po will merge. The default is to follow msgmerge's behaviour but we add some extra features with fuzzy matching:

- · If everything matches we carry that across
- We can resurrect obsolete messages for reuse
- · Messages no longer used are made obsolete
- If we cannot find a match we will first look through the current and obsolete messages and then through any global translation memory
- Fuzzy matching makes use of the Levenshtein distance algorithm to detect the best matches

# **Performance**

Fuzzy matches are usually of good quality. Installation of the python-Levenshtein package will speed up fuzzy matching. Without this a Python based matcher is used which is considerably slower.

#### **Bugs**

• pomerge and pot2po should probably become one.

# 1.3.12 csv2po

Convert between CSV (Comma Separated Value) files and the PO format. This is useful for those translators who can only use a Spreadsheet, a modern spreadsheet can open CSV files for editing. It is also useful if you have other data such as translation memory in CSV format and you wish to use it with your PO translations.

If you are starting out with your own CSV files (not created by po2csv), take note of the assumptions of the column layout explained below.

# **Usage**

```
csv2po [options] <csv> <po> po2csv [options] <po> <csv>
```

#### Where:

<csv></csv>	is a file or directory containing CSV files
<po></po>	is a file or directory containing PO files

#### Options (csv2po):

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

--progress=PROGRESS show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in csv format

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in po, pot formats

-t TEMPLATE, --template=TEMPLATE read from TEMPLATE in po, pot, pot formats

**-S, --timestamp** skip conversion if the output file has newer timestamp

**-P, --pot** output PO Templates (.pot) rather than PO files (.po)

--charset=CHARSET set charset to decode from csv files

**--columnorder=COLUMNORDER** specify the order and position of columns (location,source,target)

--duplicates=DUPLICATESTYLE what to do with duplicate strings (identical source text): merge,
msgctxt (default: 'msgctxt')

# Options (po2csv):

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

--progress=PROGRESS show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in po, pot formats

- -x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths
- -o OUTPUT, --output=OUTPUT write to OUTPUT in csv format
- -S, --timestamp skip conversion if the output file has newer timestamp
- **--columnorder=COLUMNORDER** specify the order and position of columns (location,source,target)

# **CSV** file layout

The resultant CSV file has the following layout

Col-	Data	Description
umn		
A	Location	All the PO #: location comments. These are needed to reconstruct or merge the CSV
		back into the PO file
В	Source Lan-	The msgid or source string
	guage	
С	Target Lan-	The msgstr or target language
	guage	

# **Examples**

These examples demonstrate the use of csv2po:

```
po2csv -P pot csv
```

We use the -P option to recognise POT files found in pot and convert them to CSV files placed in csv:

```
csv2po csv po
```

Convert CSV files in *csv* to PO files placed in *po*:

```
csv2po --charset=windows-1250 -t pot csv po
```

User working on Windows will often return files encoded in everything but Unicode. In this case we convert CSV files found in *csv* from *windows-1250* to UTF-8 and place the correctly encoded files in *po*. We use the templates found in *pot* to ensure that we preserve formatting and other data. Note that UTF-8 is the only available destination encoding.

```
csv2po --columnorder=location,target,source fr.csv fr.po
```

In case the CSV file has the columns in a different order you may use --columnorder.

# **Bugs**

- Translation comments #[space] and KDE comments \_: are not available in CSV mode which effects the translators effectiveness
- Locations #: that are not conformant to PO (i.e. have spaces) will get messed up by PO tools.

### 1.3.13 csv2tbx

Convert between CSV (Comma Separated Value) files and the TBX format for terminology exchange.

# **Usage**

```
csv2tbx [--charset=CHARSET] [--columnorder=COLUMNORDER] <csv> <tbx>
```

Where:

<csv></csv>	is a CSV file
<tbx></tbx>	is the target TBX file

Options (csv2tbx):

**--version** show program's version number and exit

**-h, --help** show this help message and exit

**--manpage** output a manpage based on the help

--progress=PROGRESS show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in csv format

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in tbx format

**-S, --timestamp** skip conversion if the output file has newer timestamp

--charset=CHARSET set charset to decode from csv files

**--columnorder=COLUMNORDER** specify the order and position of columns (comment,source,target)

### **CSV** file layout

The CSV file is expected to have three columns (separated by commas, not other characters like semicolons), of which the default layout is

Col-	Data	Description
umn		
A	Comment	All the PO #: location comments. These are not used in the TBX files, and can be left
		empty, but could be generated by <i>po2csv</i>
В	Source Lan-	The msgid or source string
	guage	
С	Target Lan-	The msgstr or target language
	guage	

# **Examples**

These examples demonstrate the use of csv2tbx:

```
csv2tbx terms.csv terms.tbx
```

to simply convert terms.csv to terms.tbx.

To convert a directory recursively to another directory with the same structure of files:

```
csv2tbx csv-dir tbx-target-dir
```

This will convert CSV files in csv-dir to TBX files placed in tbx-target-dir.:

```
csv2tbx --charset=windows-1250 csv tbx
```

Users working on Windows will often return files in encoding other the Unicode based encodings. In this case we convert CSV files found in *csv* from *windows-1250* to UTF-8 and place the correctly encoded files in *tbx*. Note that UTF-8 is the only available destination encoding.

### Two column CSV

```
csv2tbx --columnorder=source,target foo.csv foo.tbx
```

#### **Notes**

For conformance to the standards and to see which features are implemented, see CSV and TBX.

# 1.3.14 tbx2po

Convert between TermBase eXchange (.tbx) glossary format and Gettext PO format.

### **Usage**

```
tbx2po <tbx> <po>
```

Where:

<tbx></tbx>	is a TBX file
<po></po>	is the target PO file

# Options (tbx2po):

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

--progress=PROGRESS show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in csv format

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in tbx format

**-S, --timestamp** skip conversion if the output file has newer timestamp

# **Examples**

These examples demonstrate the use of tbx2po:

```
tbx2po terms.tbx terms.po
```

to simply convert terms.tbx to terms.po.

To convert a directory recursively to another directory with the same structure of files:

```
tbx2po tbx-dir po-target-dir
```

This will convert TBX files in *tbx-dir* to PO files placed in *po-target-dir*.

### **Notes**

For conformance to the standards and to see which features are implemented, see PO Files and TBX.

# 1.3.15 html2po

Convert translatable items in HTML to the PO format.

# **Usage**

```
html2po [options] <html> <po>po2html [options] <po> <html>
```

Where:

<html></html>	is an HTML file or a directory of HTML files
<po></po>	is a PO file or directory of PO files

Options (html2po):

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

**--progress=PROGRESS** show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in htm, html, xhtml formats

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in po, pot formats

-S, --timestamp skip conversion if the output file has newer timestamp
-P, --pot output PO Templates (.pot) rather than PO files (.po)

**-u, --untagged** include untagged sections

**--keepcomments** preserve html comments as translation notes in the output

--duplicates=DUPLICATESTYLE what to do with duplicate strings (identical source text): merge,
msgctxt (default: 'msgctxt')

Options (po2html):

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

--progress=PROGRESS show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in po, pot formats

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in htm, html, xhtml formats

-t TEMPLATE, --template=TEMPLATE read from TEMPLATE in htm, html, xhtml formats

**-S, --timestamp** skip conversion if the output file has newer timestamp

--threshold=PERCENT only convert files where the translation completion is above PERCENT

**--fuzzy** use translations marked fuzzy

**--nofuzzy** don't use translations marked fuzzy (default)

# **Examples**

```
html2po -P site pot
```

This will find all HTML files (.htm, .html, .xhtml) in site, convert them to POT files and place them in pot:

```
po2html -t site xh site-xh
```

All the PO translations in xh will be converted to html using html files in *site* as templates and outputting new translated HTML files in *site-xh* 

# **Bugs**

We don't hide enough of some of the tags, e.g. <a> tags have too much exposed, we should expose only what needs to be translated and allow the changing on position of the tag within the translation block. Similarly there is some markup that could be excluded e.g. <b> tags that appear at the start and end of a msgid, i.e. they don't need placement from the translator.

If the HTML is indented you get very odd msgid's

Some items end up in the msgid's that should not be translated

It might be worth investigating http://opensource.bureau-cornavin.com/html2pot-po2html/index.html which uses XSLT to transform XHTML to Gettext PO

# 1.3.16 flatxml2po

Converts flat XML (.xml) files to Gettext PO format, a simple monolingual and single-level XML.

### Usage

```
flatxml2po [options] <xml> <po>
po2flatxml [options] <po> <xml> [-t <base-xml>]
```

#### Where:

<xml></xml>	is a valid .xml file or directory of those files
<po></po>	is a directory of PO or POT files
<base-< td=""><td>is a template or the original file before translation. required for roundtrips preserving extraneous</td></base-<>	is a template or the original file before translation. required for roundtrips preserving extraneous
xml>	data.

## Options (flatxml2po):

**--version** show program's version number and exit

**-h, --help** show this help message and exit

--manpage output a manpage based on the help

--progress=PROGRESS show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in xml format

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in po, pot formats

**-S, --timestamp** skip conversion if the output file has newer timestamp

-r ROOT, --root=ROOT name of the XML root element (default: "root")

-v VALUE, --value=VALUE name of the XML value element (default: "str")

-k KEY, --key=KEY name of the XML key attribute (default: "key")

-n NS, --namespace=NS XML namespace uri (default: None)

### Options (po2flatxml):

**--version** show program's version number and exit

**-h, --help** show this help message and exit

**--manpage** output a manpage based on the help

**--progress=PROGRESS** show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in po, pot formats

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in xml format

-t TEMPLATE, --template=TEMPLATE read from TEMPLATE in xml format

-S, --timestamp skip conversion if the output file has newer timestamp

**-r ROOT, --root=ROOT** name of the XML root element (default: "root")

-v VALUE, --value=VALUE name of the XML value element (default: "str")

-k KEY, --key=KEY name of the XML key attribute (default: "key")

- -n NS, --namespace=NS XML namespace uri (default: None)
- -w INDENT, --indent=INDENT indent width in spaces, 0 for no indent (default: 2)

## **Formats Supported**

Check *flat XML format* document to see to which extent the XML format is supported.

### **Examples**

This example looks at roundtrip of flat XML translations as well as recovery of existing translations.

First we need to create a set of POT files.:

```
flatxml2po -P lang/en pot/
```

All .xml files found in the lang/en directory are converted to Gettext POT files and placed in the pot directory.

If you are translating for the first time then you can skip the next step. If you need to recover your existing translations then we do the following:

```
flatxml2po -t lang/en lang/zu po-zu/
```

Using the English XML files found in lang/en and your existing Zulu translation in lang/zu we create a set of PO files in po-zu. These will now have your translations. Please be aware that in order for that to work 100% you need to have both English and Zulu at the same revision, if they are not you will have to review all translations.

You are now in a position to translate your recovered translations or your new POT files.

Once translated you can convert back as follows:

```
po2flatxml -t lang/en po-zu/ lang/zu
```

Your translations found in the Zulu PO directory, po-zu, will be converted to XML using the files in lang/en as templates and placing your new translations in lang/zu.

To update your translations simply redo the POT creation step and make use of *pot2po* to bring your translation up-to-date.

### Limitations

Indentation only supports spaces (specified with --indent greater than zero) or flattened (no indentation, everything on a single line; specified with --indent set to zero). Tabs are not supported using po2flatxml.

# 1.3.17 ical2po

New in version 1.2.

Converts iCalendar (\*.ics) files to Gettext PO format.

### Usage

```
ical2po [options] <ical> <po>
po2ical [options] -t <ical> <po> <ical>
```

#### Where:

<ical></ical>	is a valid .ics file or directory of those files
<po></po>	is a directory of PO or POT files

# Options (ical2po):

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

**--progress=PROGRESS** show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in ics format

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in po, pot formats

-t TEMPLATE, --template=TEMPLATE read from TEMPLATE in ics format

-S, --timestamp skip conversion if the output file has newer timestamp
-P, --pot output PO Templates (.pot) rather than PO files (.po)

--duplicates=DUPLICATESTYLE what to do with duplicate strings (identical source text): merge,
msgctxt (default: 'msgctxt')

### Options (po2ical):

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

**--progress=PROGRESS** show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in po, pot formats

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in ics format

-t TEMPLATE, --template=TEMPLATE read from TEMPLATE in ics format

-S, --timestamp skip conversion if the output file has newer timestamp

--threshold=PERCENT only convert files where the translation completion is above PERCENT

**--fuzzy** use translations marked fuzzy

**--nofuzzy** don't use translations marked fuzzy (default)

### **Examples**

This example looks at roundtrip of iCalendar translations. While you can do recovery of translations, its unlikely that you will ever need to do that.

First we need to create a set of POT files.

```
ical2po -P ical.ics ical.pot
```

The ical.ics file is converted to Gettext POT files called ical.pot. Directories of iCalendar files can also be processed.

Begin translating the ical.pot file by first copying it to make a PO file.

```
cp ical.pot ical-af.po
```

You are now in a position to translate the file ical-af.po in your favourite translation tool.

Once translated you can convert back as follows:

```
po2ical -t ical.ics ical-af.po ical-af.ics
```

Your translations found in the Afrikaans PO file, ical-af.po, will be converted to .ics using the file ical.ics as a template and creating your newly translated .ics file ical-af.ics.

To update your translations simply redo the POT creation step and make use of *pot2po* to bring your translation up-to-date.

#### **Notes**

The converter will only process events in the calendar file, the file itself can contain many other things that could be localisable. Please raise a bug if you want to extract additional items.

The converter does not make use of the LANGUAGE attribute which is permitted in the format. The LANGUAGE attribute does not aid multilingualism in this context so is ignored.

The converter could conceivably also process vCard files, but this has not been implemented for lack of a clear need. Please raise a bug with an example if you have such a file that could benefit from localisation.

# 1.3.18 ini2po

Converts .ini files to Gettext PO format.

### **Usage**

```
ini2po [options] <ini> <po>
po2ini [options] -t <ini> <po> <ini>
```

Where:

<ini></ini>	is a valid .ini file or directory of those files
<po></po>	is a directory of PO or POT files

Options (ini2po):

**--version** show program's version number and exit

```
-h, --help
                           show this help message and exit
      --manpage
                           output a manpage based on the help
      --progress=PROGRESS show progress as: dots, none, bar, names, verbose
      --errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback
      -i INPUT, --input=INPUT read from INPUT in ini, isl, iss formats
      -x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths
      -o OUTPUT, --output=OUTPUT write to OUTPUT in po, pot formats
      -t TEMPLATE, --template=TEMPLATE read from TEMPLATE in ini, isl, iss formats
      -S, --timestamp
                           skip conversion if the output file has newer timestamp
      -P, --pot
                           output PO Templates (.pot) rather than PO files (.po)
      --duplicates=DUPLICATESTYLE what to do with duplicate strings (identical source text): merge,
                           msgctxt (default: 'msgctxt')
Options (po2ini):
      --version
                           show program's version number and exit
      -h, --help
                           show this help message and exit
                           output a manpage based on the help
      --manpage
      --progress=PROGRESS show progress as: dots, none, bar, names, verbose
      --errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback
      -i INPUT, --input=INPUT read from INPUT in po, pot formats
      -x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths
      -o OUTPUT, --output=OUTPUT write to OUTPUT in ini, isl formats
      -t TEMPLATE, --template=TEMPLATE read from TEMPLATE in ini, isl formats
      -S, --timestamp
                           skip conversion if the output file has newer timestamp
      --threshold=PERCENT only convert files where the translation completion is above PERCENT
      --fuzzy
                           use translations marked fuzzy
      --nofuzzy
                           don't use translations marked fuzzy (default)
```

## **Formats Supported**

INI files need to be organized into separate languages per file and in the following format:

```
[Section]
; a comment
a = a string
```

Comment marked with the hash symbol (#) are also allowed, and the colon (:) is also accepted as key-value delimiter:

```
[Section]
# another comment
b: a string
```

This variants in comment marks and key-value delimiters can be mixed in one single INI file:

```
[Section]
; a comment
a = a string
# another comment
b: a string
c:'other example with apostrophes'
d:"example with double quotes"
```

The spacing between the key-value delimiter and the key, and the between the value and the key-value delimiter is not important since the converter automatically strips the blank spaces.

**Note:** A section must be present at the file beginning in order to get ini2po working properly. You may add it by hand at the file beginning.

**Note:** Strings marked with double quotes and/or apostrophes will carry these quotation marks to the generated .po file, so they will appear like:

```
#: [Section]c
msgid "'other example with apostrophes'"
msgstr ""

#: [Section]d
msgid "\"example with double quotes\""
msgstr ""
```

### **Examples**

This example looks at roundtrip of .ini translations as well as recovery of existing translations.

First we need to create a set of POT files.

```
ini2po -P ini/ pot/
```

All .ini files found in the ini/ directory are converted to Gettext POT files and placed in the pot/ directory.

If you are translating for the first time then you can skip the next step. If you need to recover your existing translations then we do the following:

```
ini2po -t lang/ zu/ po-zu/
```

Using the English .ini files found in lang/ and your existing Zulu translation in zu/ we create a set of PO files in po-zu/. These will now have your translations. Please be aware that in order for the to work 100% you need to have both English and Zulu at the same revision. If they are not, you will have to review all translations.

You are now in a position to translate your recovered translations or your new POT files.

Once translated you can convert back as follows:

```
po2ini -t lang/ po-zu/ zu/
```

Your translations found in the Zulu PO directory, po-zu/, will be converted to .ini using the files in lang/ as templates and placing your newly translated .ini files in zu/.

To update your translations simply redo the POT creation step and make use of pot2po to bring your translation up-to-date.

### **Issues**

We do not extract comments from .ini files. These are sometimes needed as developers provide guidance to translators in these comments.

# 1.3.19 json2po

Converts .json files to Gettext PO format.

### **Usage**

```
json2po [options] <json> <po>
po2json [options] -t <json> <po> <json>
```

### Where:

<json></json>	is a valid .json file or directory of those files
<po></po>	is a directory of PO or POT files

### Options (json2po):

```
--version
                    show program's version number and exit
-h, --help
                    show this help message and exit
                    output a manpage based on the help
--manpage
--progress=PROGRESS show progress as: dots, none, bar, names, verbose
--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback
-i INPUT, --input=INPUT read from INPUT in JSON format
-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths
-o OUTPUT, --output=OUTPUT write to OUTPUT in po, pot formats
-t TEMPLATE, --template=TEMPLATE read from TEMPLATE in JSON format
-S, --timestamp
                    skip conversion if the output file has newer timestamp
-P, --pot
                    output PO Templates (.pot) rather than PO files (.po)
--filter=FILTER
                    leaves to extract e.g. 'name,desc': (default: extract everything)
--duplicates=DUPLICATESTYLE what to do with duplicate strings (identical source text): merge,
                    msgctxt (default: 'msgctxt')
```

Options (po2json):

```
--version
                     show program's version number and exit
-h, --help
                     show this help message and exit
--manpage
                     output a manpage based on the help
--progress=PROGRESS show progress as: dots, none, bar, names, verbose
```

- --errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback
- -i INPUT, --input=INPUT read from INPUT in po, pot formats
- -x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths
- -o OUTPUT, --output=OUTPUT write to OUTPUT in JSON format
- -t TEMPLATE, --template=TEMPLATE read from TEMPLATE in JSON format
- -S, --timestamp skip conversion if the output file has newer timestamp
- --threshold=PERCENT only convert files where the translation completion is above PERCENT
- **--fuzzy** use translations marked fuzzy
- **--nofuzzy** don't use translations marked fuzzy (default)
- **--removeuntranslated** remove untranslated strings from output

### **Examples**

This example looks at roundtrip of .json translations as well as recovery of existing translations.

First we need to create a set of POT files.

```
json2po -P json/ pot/
```

All .json files found in the json/directory are converted to Gettext POT files and placed in the pot/directory.

If you are translating for the first time then you can skip the next step. If you need to recover your existing translations then we do the following:

```
json2po -t lang/ zu/ po-zu/
```

Using the English .json files found in lang/ and your existing Zulu translation in zu/ we create a set of PO files in po-zu/. These will now have your translations. Please be aware that in order for the to work 100% you need to have both English and Zulu at the same revision. If they are not, you will have to review all translations.

You are now in a position to translate your recovered translations or your new POT files.

Once translated you can convert back as follows:

```
po2json -t lang/ po-zu/ zu/
```

Your translations found in the Zulu PO directory, po-zu/, will be converted to .json using the files in lang/ as templates and placing your newly translated .json files in zu/.

To update your translations simply redo the POT creation step and make use of *pot2po* to bring your translation up-to-date.

# 1.3.20 web2py2po

Converts web2py translation files to PO files and vice versa.

Web2py, formerly known as Gluon) is an open-source, Python-based web application framework by Massimo Di Pierro (inspired by Django and Rails).

Web2py uses an internal localization engine based on Python dictionaries, which is applied with the T() lookup function. Web2py provides a built-in translation interface for the T()-engine, which is excellent for rapid application development.

On the other hand, for collaboration and workflow control in a wider community you might probably rather want to use Pootle, Launchpad or similar facilities for translation, thus need to transform the web2py dictionaries into PO files and vice versa. And exactly that is what the web2py2po converters are good for.

## **Usage**

```
web2py2po [options] <web2py> <po>
po2web2py [options] <po> <web2py>
```

### Where:

<web2py></web2py>	is a valid web2py translation file
<po></po>	is a PO or POT file or a directory of PO or POT files

### Options (web2py2po):

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

--progress=PROGRESS show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in php format

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in po, pot formats

-S, --timestamp skip conversion if the output file has newer timestamp
-P, --pot output PO Templates (.pot) rather than PO files (.po)

--duplicates=DUPLICATESTYLE what to do with duplicate strings (identical source text): merge,
msgctxt (default: 'msgctxt')

# Options (po2web2py):

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

**--progress=PROGRESS** show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in po, pot formats

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in php format

**-S, --timestamp** skip conversion if the output file has newer timestamp

--threshold=PERCENT only convert files where the translation completion is above PERCENT

**--fuzzy** use translations marked fuzzy

**--nofuzzy** don't use translations marked fuzzy (default)

### **Notes**

### Handling of blanks/untranslated messages:

Untranslated messages in the web2py translation files are usually marked with a leading %%"\*\*\* "%%, so:

- All target strings from the web2py sources with a leading %%"\*\*\* "%% are inserted as blank msgstr's into the PO result (web2py2po)
- Blank msgstr's from the PO file will get the msgid string with a leading %%"\*\*\* "%% as target string in the web2py result (po2web2py)

# 1.3.21 rc2po

Converts Windows Resource .rc files to Gettext PO format.

### **Usage**

```
rc2po [options] <rc> <po> po2rc [options] -t <rc> <po> <rc>
```

#### Where:

<rc></rc>	is a valid Windows Resource file or directory of those files	
<po></po>	<po> is a directory of PO or POT files</po>	

### Options (rc2po):

```
--version
                    show program's version number and exit
-h, --help
                    show this help message and exit
--manpage
                    output a manpage based on the help
--progress=PROGRESS show progress as: dots, none, bar, names, verbose
--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback
-i INPUT, --input=INPUT read from INPUT in rc format
-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths
-o OUTPUT, --output=OUTPUT write to OUTPUT in po, pot formats
-t TEMPLATE, --template=TEMPLATE read from TEMPLATE in rc format
-S, --timestamp
                    skip conversion if the output file has newer timestamp
-P, --pot
                    output PO Templates (.pot) rather than PO files (.po)
--charset=CHARSET charset to use to decode the RC files (default: cp1252)
-l LANG, --lang=LANG LANG entry (default: LANG_ENGLISH)
--sublang=SUBLANG SUBLANG entry (default: SUBLANG_DEFAULT)
--duplicates=DUPLICATESTYLE what to do with duplicate strings (identical source text): merge,
                    msgctxt (default: 'msgctxt')
```

**--version** show program's version number and exit

```
-h, --help show this help message and exit
```

**--manpage** output a manpage based on the help

- --progress=PROGRESS show progress as: dots, none, bar, names, verbose
- --errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback
- -i INPUT, --input=INPUT read from INPUT in po, pot formats
- -x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths
- -o OUTPUT, --output=OUTPUT write to OUTPUT in rc format
- -t TEMPLATE, --template=TEMPLATE read from TEMPLATE in rc format
- **-S, --timestamp** skip conversion if the output file has newer timestamp
- --charset=CHARSET charset to use to decode the RC files (default: utf-8)
- -l LANG, --lang=LANG LANG entry
- --sublang=SUBLANG SUBLANG entry (default: SUBLANG\_DEFAULT)
- --threshold=PERCENT only convert files where the translation completion is above PERCENT
- **--fuzzy** use translations marked fuzzy
- **--nofuzzy** don't use translations marked fuzzy (default)

## **Formats Supported**

**Note:** This implementation is based mostly on observing WINE .rc files, these should mimic other non-WINE .rc files.

### **Examples**

This example looks at roundtrip of Windows Resource translations as well as recovery of existing translations.

First we need to create a set of POT files.

```
rc2po -P lang/ pot/
```

All .rc files found in the lang/directory are converted to Gettext POT files and placed in the pot/directory.

If you are translating for the first time then you can skip the next step. If you need to recovery your existing translations then we do the following:

```
rc2po -t lang zu po-zu/
```

Using the English .rc files found in lang and your existing Zulu translation in zu we create a set of PO files in po-zu. These will now have your translations. Please be aware that in order for the to work 100% you need to have both English and Zulu at the same revision, if they are not you will have to review all translations. Also the .rc files may be in different encoding, we cannot at the moment process files of different encodings and assume both are in the same encoding supplied.

You are now in a position to translate your recovered translations or your new POT files.

Once translated you can convert back as follows:

```
po2rc -t lang/ po-zu/ zu/
```

Your translations found in the Zulu PO directory, po-zu, will be converted to .rc using the files in lang/ as templates and placing your new translations in zu/.

To update your translations simply redo the POT creation step and make use of *pot2po* to bring your translation up-to-date.

#### **Issues**

If you are recovering translation using rc2po -t en.rc xx.rc xx.po then both en.rc and xx.rc need to be in the same encoding.

There might be problems with MENUs that are deaply nested.

# 1.3.22 resx2po

Converts .Net Resource (.resx) files to Gettext PO format, a monolingual file format used in Microsoft .Net Applications.

# **Usage**

```
resx2po [options] <resx> <po>
po2resx [options] <po> <resx> -t <resx>
```

### Where:

<resx></resx>	is a valid .resx file or directory of those files
<po></po>	is a directory of PO or POT files

### Options (resx2po):

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

--progress=PROGRESS show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in RESX format

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in po, pot formats

-t TEMPLATE, --template=TEMPLATE read from TEMPLATE in RESX format

-S, --timestamp skip conversion if the output file has newer timestamp
-P, --pot output PO Templates (.pot) rather than PO files (.po)

--filter=FILTER leaves to extract e.g. 'name,desc': (default: extract everything)

**--duplicates=DUPLICATESTYLE** what to do with duplicate strings (identical source text): merge, msgctxt (default: 'msgctxt')

Options (po2resx):

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

--progress=PROGRESS show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in po, pot formats

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in RESX format

-t TEMPLATE, --template=TEMPLATE read from TEMPLATE in RESX format

**-S, --timestamp** skip conversion if the output file has newer timestamp

**--fuzzy** use translations marked fuzzy

**--nofuzzy** don't use translations marked fuzzy (default)

### **Examples**

This example looks at roundtrip of .resx translations as well as recovery of existing translations.

First we need to create a set of POT files

```
resx2po -P resx/ pot/
```

All .resx files found in the resx/ directory are converted to Gettext POT files and placed in the pot/ directory.

If you are translating for the first time then you can skip the next step. If you need to recover your existing translations then we do the following

```
resx2po zu/ po-zu/ -t lang/
```

Using the English .resx files found in lang/ and your existing Zulu translation in zu/ we create a set of PO files in po-zu/. These will now have your translations. Please be aware that in order for the to work 100% you need to have both English and Zulu at the same revision. If they are not, you will have to review all translations.

You are now in a position to translate your recovered translations or your new POT files.

Once translated you can convert back as follows:

```
po2resx po-zu/ zu/ -t lang/
```

Your translations found in the Zulu PO directory, po-zu/, will be converted to .resx using the files in lang/ as templates and placing your newly translated .resx files in zu/.

To update your translations simply redo the POT creation step and make use of *pot2po* to bring your translation up-to-date.

# 1.3.23 symb2po

New in version 1.3.

Converts Symbian-style translation files to PO files and vice versa. The Symbian translation files currently have a strong Buddycloud flavour, but the tools will be made more general as the need arises.

### **Usage**

```
symb2po [options] [-t <target_lang_symb>] <source_lang_symb> <po>
po2symb [options] -t <target_lang_symb> <po> <target_lang_symb>
```

### Where:

<target_lang_symb></target_lang_symb>	is a valid Symbian translation file or directory of those files
<source_lang_symb></source_lang_symb>	is a valid Symbian translation file or directory of those files
<po></po>	is a PO or POT file or a directory of PO or POT files

### Options (symb2po):

**--version** show program's version number and exit

**-h, --help** show this help message and exit

**--manpage** output a manpage based on the help

--progress=PROGRESS show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in php format

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in po, pot formats

-t TEMPLATE, --template=TEMPLATE read from TEMPLATE in the Symbian translation format

**-S, --timestamp** skip conversion if the output file has newer timestamp

**-P, --pot** output PO Templates (.pot) rather than PO files (.po)

--duplicates=DUPLICATESTYLE what to do with duplicate strings (identical source text): merge,
msgctxt (default: 'msgctxt')

# Options (po2symb):

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

--progress=PROGRESS show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in po, pot formats

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in php format

-t TEMPLATE, --template=TEMPLATE read from TEMPLATE in the Symbian translation format

**-S, --timestamp** skip conversion if the output file has newer timestamp

## **Examples**

### symb2po

The most common use of symb2po, is to generate a POT (PO template) file from the English translation (note that the tool currently expects the Symbian translation file to end with the extension .r01, which is the code for English translation files). This file then serves as the source document from which all translations will be derived.

To create a POT file called my\_project.pot from the source Symbian translation file my\_project.ro1, the following is executed:

```
symb2po my_project.r01 my_project.pot
```

In order to re-use existing translations in the Symbian translation format, symb2po can merge that translation into the source Symbian translation to produce a translated PO file. The existing Symbian translation file is specified with the -t flag.

To create a file called my\_project-en-fr.po (this is not the recommended PO naming convention) from the source Symbian translation file my\_project.r01 and its French translation my\_project.r02, execute:

```
{\tt symb2po-t\ my\_project.r02\ my\_project.r01\ my\_project-en-fr.po}
```

**Note:** Ensure that the English and French files are well aligned, in other words, no changes to the source text should have happened since the translation was done.

### po2symb

The po2symb tool is used to extract the translations in a PO into a template Symbian translation file. The template Symbian translation file supplies the "shape" of the generated file (formatting and comments).

In order to produce a French Symbian translation file using the English Symbian translation file my\_project.r01 as a template and the PO file my\_project-en-fr.po (this is not the recommended PO naming convention) as the source document, execute:

```
po2symb -t my_project.r01 my_project-en-fr.po my_project.r02
```

### **Notes**

The tools won't touch anything appearing between lines marked as:

```
// DO NOT TRANSLATE
```

The string r\_string\_languagegroup\_name is used to set the Language-Team PO header field.

The Symbian translation header field Author is used to set the Last-Translator PO header field.

### Issues

The file format is heavily tilted towards the Buddycould implementation

The tools do nothing with the Name and Description Symbian header fields. This means that po2symb will just copy the values in the supplied template. So you might see something such as:

```
Description : Localisation File : English
```

in a generated French translation file.

### **Bugs**

Probably many, since this software hasn't been tested much yet.

# 1.3.24 tiki2po

Converts TikiWiki language.php files to Gettext PO format.

### **Usage**

```
tiki2po [options] <tiki> <po>po2tiki [options] <po> <tiki>
```

#### Where:

<tiki></tiki>	is a valid language.php file for TikiWiki
<po></po>	is a PO file

## Options (tiki2po):

**--version** show program's version number and exit

**-h, --help** show this help message and exit

**--manpage** output a manpage based on the help

**--progress=PROGRESS** show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in php format

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in po, pot formats

**-S, --timestamp** skip conversion if the output file has newer timestamp

**--include-unused** When converting, include strings in the "unused" section?

### Options (po2tiki):

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

**--progress=PROGRESS** show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in po, pot formats

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

```
-o OUTPUT, --output=OUTPUT write to OUTPUT in php format
```

**-S, --timestamp** skip conversion if the output file has newer timestamp

## **Examples**

These examples demonstrate the use of tiki2po:

```
tiki2po language.po
```

Convert the tiki language.php file to .po:

```
po2tiki language.po language.php
```

Convert a .po file to a tiki language.php file

### **Notes**

• Templates are not currently supported.

# 1.3.25 ts2po

Convert Qt .ts localization files to Gettext .po format files using ts2po and convert the translated *PO Files* files back to *Qt .ts* using po2ts.

The Qt toolkit comes with a localization application, Qt Linguist, however you might wish to standardise on one localization tool. ts2po allows you to standardise on the PO format and PO related tools.

Note: Virtaal and Pootle can edit .ts files directly without the need for any conversion.

**Warning:** po2ts uses our older .ts support. Thus many of the newer features in .ts are not supported. To support those features rather edit directly in Virtaal or Pootle.

### **Usage**

```
ts2po [options] <ts> <po> po2ts [options] <po> <ts>
```

Where:

<ts></ts>	is a Qt .ts file or directory that contains .ts files
<po></po>	is a PO file or a directory of PO files

Options (ts2po):

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

- **--progress=PROGRESS** show progress as: dots, none, bar, names, verbose
- --errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback
- -i INPUT, --input=INPUT read from INPUT in ts format
- -x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths
- -o OUTPUT, --output=OUTPUT write to OUTPUT in po, pot formats
- **-S, --timestamp** skip conversion if the output file has newer timestamp
- **-P, --pot** output PO Templates (.pot) rather than PO files (.po)
- --duplicates=DUPLICATESTYLE what to do with duplicate strings (identical source text): *merge*, *msgctxt* (default: 'msgctxt')

### Options (po2ts):

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

- --progress=PROGRESS show progress as: dots, none, bar, names, verbose
- --errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback
- -i INPUT, --input=INPUT read from INPUT in po, pot formats
- -x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths
- -o OUTPUT, --output=OUTPUT write to OUTPUT in ts format
- -t TEMPLATE, --template=TEMPLATE read from TEMPLATE in ts format
- **-S, --timestamp** skip conversion if the output file has newer timestamp
- -c CONTEXT, --context=CONTEXT use supplied context instead of the one in the .po file comment

## **Examples**

```
ts2po -P psi.ts psi.pot
```

This will create a POT file called *psi.pot* from the Qt .ts file called *psi.ts*.

```
po2ts af.po psi_af.ts
```

Now take your translated PO files af.po and convert it into a translated Qt .ts file, psi\_af.ts.

**Note:** You need to use the tools from the Qt toolkit to create the compiled .qm language files for the application.

## **Bugs**

There are probably still some bugs related to migrating the various attributes across for the different formats. The converters don't support all the newer features of the TS format, whereas the native support of Virtaal and Pootle is much better.

# 1.3.26 xliff2po

Converts XLIFF localization files to Gettext PO files. XLIFF is the XML Localization Interchange File Format developed by OASIS (Organization for the Advancement of Structured Information Standards) to allow translation work to be standardised no matter what the source format and to allow the work to be freely moved from tool to tool.

### **Usage**

```
po2xliff [options] <po> <xliff>
xliff2po [options] <xliff> <po>
```

### Where:

<po></po>	is a PO file or directory of PO files
<xliff></xliff>	is an XLIFF file or directory of XLIFF files

# Options (xliff2po):

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

--progress=PROGRESS show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in xliff format

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in po, pot formats

-S, --timestamp skip conversion if the output file has newer timestamp
-P, --pot output PO Templates (.pot) rather than PO files (.po)

--duplicates=DUPLICATESTYLE what to do with duplicate strings (identical source text): merge,
msgctxt (default: 'msgctxt')

# Options (po2xliff):

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

--progress=PROGRESS show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in po, pot formats

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in xliff format

-t TEMPLATE, --template=TEMPLATE read from TEMPLATE in xliff format

-S, --timestamp skip conversion if the output file has newer timestamp

### **Examples**

```
xliff2po -P xliff pot
```

Create POT files from the XLIFF files found in directory xliff and output them to the directory pot

```
po2xliff xh xh-xlf
```

Convert the Xhosa PO files in xh to XLIFF and place them in xh-xlf

### **Bugs**

This filter is not yet extensively used... expect bugs. See *XLIFF* to see how well our implementation conforms to the standard.

The PO plural implementation is still very new and needs active testing.

# 1.3.27 I20n2po

Convert between L20n files (.ftl) and Gettext PO format.

### **Usage**

```
120n2po [options] <ftl> <po>po2120n [options] -t <template> <po> <ftl>
```

### Where:

<ftl></ftl>	is a directory containing ftl files or an individual ftl file
<po></po>	is a directory containing PO files and an individual ftl file
<template></template>	is a directory of template ftl files or a single template ftl file

## Options (120n2po):

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

--progress=PROGRESS show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in 120n format

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in po, pot formats

**-t TEMPLATE**, **--template=TEMPLATE** read from TEMPLATE in 120n format

-S, --timestamp skip conversion if the output file has newer timestamp
-P, --pot output PO Templates (.pot) rather than PO files (.po)

--duplicates=DUPLICATESTYLE what to do with duplicate strings (identical source text): *merge*, *msgctxt* (default: 'msgctxt')

Options (po2l20n):

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

--progress=PROGRESS show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in po, pot formats

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in 120n format

**-t TEMPLATE**, **--template=TEMPLATE** read from TEMPLATE in 120n format

**-S, --timestamp** skip conversion if the output file has newer timestamp

--removeuntranslated remove key value from output if it is untranslated

--threshold=PERCENT only convert files where the translation completion is above PERCENT

**--fuzzy** use translations marked fuzzy

**--nofuzzy** don't use translations marked fuzzy (default)

## **Examples**

These examples demonstrate most of the useful invocations of 120n2po:

# **Creating POT files**

```
120n2po -P 120n pot
```

Extract messages from l20n directory and place them in a directory called pot. The -P option ensures that we create POT files instead of PO files.:

```
120n2po -P file.ftl file.pot
```

Extract messages from file.ftl and place them in file.pot.

## Creating PO files from existing work

```
120n2po --duplicates=msgctxt -t reference zu zu-po
```

Extract all existing Zulu messages from *zu* directory and place the resultant PO files in a directory called *zu-po*. If you find duplicate messages in a file then use Gettext's mgsctxt to disambiguate them. During the merge we use the .ftl files in *reference* as templates and as the source of the English text for the msgid. Once you have your PO files you might want to use *pomigrate2* to ensure that your PO files match the latest POT files.

### Creating .ftl files from your translations

```
po2120n -t reference zu-po zu
```

Using our translations found in zu-po and the templates found in reference we create a new set of ftl files in zu. These new ftl files will look exactly like those found in the templates, but with the text changed to the translation. Any fuzzy entry in our PO files will be ignored and any untranslated item will be placed in zu in English.

# 1.3.28 yaml2po

New in version 2.2.6.

Converts YAML localization files to Gettext PO format.

### **Usage**

```
yaml2po [options] <yml> <po>po2yaml [options] <po> <yml>
```

#### Where:

<yml></yml>	is a valid YAML localisable file or directory of those files
<po></po>	is a directory of PO or POT files

# Options (yaml2po):

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

--progress=PROGRESS show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in yaml, yml formats

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in po, pot formats

-t TEMPLATE, --template=TEMPLATE read from TEMPLATE in yaml, yml formats

-S, --timestamp skip conversion if the output file has newer timestamp
-P, --pot output PO Templates (.pot) rather than PO files (.po)

--duplicates=DUPLICATESTYLE what to do with duplicate strings (identical source text): merge,
msgctxt (default: 'msgctxt')

# Options (po2yaml):

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

--progress=PROGRESS show progress as: dots, none, bar, names, verbose

```
--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback
```

- -i INPUT, --input=INPUT read from INPUT in po, pot formats
- -x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths
- -o OUTPUT, --output=OUTPUT write to OUTPUT in yaml, yml formats
- -t TEMPLATE, --template=TEMPLATE read from TEMPLATE in yaml, yml formats
- **-S, --timestamp** skip conversion if the output file has newer timestamp
- --threshold=PERCENT only convert files where the translation completion is above PERCENT
- **--fuzzy** use translations marked fuzzy
- **--nofuzzy** don't use translations marked fuzzy (default)

## **Formats Supported**

Check YAML format document to see to which extent the YAML format is supported.

### **Examples**

This example looks at roundtrip of YAML translations as well as recovery of existing translations.

First we need to create a set of POT files:

```
yaml2po -P lang/en pot/
```

All .yml files found in the lang/en directory are converted to Gettext POT files and placed in the pot directory.

If you are translating for the first time then you can skip the next step. If you need to recover your existing translations then we do the following:

```
yaml2po -t lang/en lang/zu po-zu/
```

Using the English YAML files found in lang/en and your existing Zulu translation in lang/zu we create a set of PO files in po-zu. These will now have your translations. Please be aware that in order for that to work 100% you need to have both English and Zulu at the same revision, if they are not you will have to review all translations.

You are now in a position to translate your recovered translations or your new POT files.

Once translated you can convert back as follows:

```
po2yaml -t lang/en po-zu/ lang/zu
```

Your translations found in the Zulu PO directory, po-zu, will be converted to YAML using the files in lang/en as templates and placing your new translations in lang/zu.

To update your translations simply redo the POT creation step and make use of *pot2po* to bring your translation up-to-date.

### 1.3.29 -errorlevel=ERRORLEVEL

This is a parameter that can be passed to most of the programs in the translate toolkit in order to choose the level of feedback that you need when errors occur. It is mostly useful for debugging. Please report your errors to the developers with --errorlevel=traceback.

#### none

Display no error messages

### message

Display on the error message

```
An error occurred processing PO file
```

### exception

Give the error message and name and Python exception

```
ValueError: An error occurred processing PO file
```

#### traceback

Provide a full traceback for debugging purposes

```
csv2po: warning: Error processing: nso/readlicense_oo/docs/readme.csv: Traceback_
→ (most recent call last):
 File "/usr/lib/python2.4/site-packages/translate/misc/optrecurse.py", line 415, in_
\rightarrowrecursiveprocess
    success = self.processfile(fileprocessor, options, fullinputpath, fulloutputpath, __
→fulltemplatepath)
 File "/usr/lib/python2.4/site-packages/translate/misc/optrecurse.py", line 468, in.
→processfile
   if fileprocessor(inputfile, outputfile, templatefile, **passthroughoptions):
 File "/usr/lib/python2.4/site-packages/translate/convert/csv2po.py", line 183, in_
\rightarrowconvertcsv
   outputpo = convertor.convertfile(inputcsv)
 File "/usr/lib/python2.4/site-packages/translate/convert/csv2po.py", line 159, in_
→convertfile
    raise ValueError("An error occurred processing PO file")
ValueError: An error occurred processing PO file
```

# 1.3.30 -duplicates=DUPLICATESTYLE

Gettext PO files only allow one message with a common msgid (source string). Many other formats allow duplicate entries. To create a valid PO file you need to merge these duplicate entries into one PO message. However, this often negatively affects the roundtrip or is not what is expected by the user. Thus we have a number of methods of handling duplicates which we call *duplicate styles*.

Also affected are conversions in which the source format is empty (allowing possible translation). As the header in a PO file is identified by an empty source string, your message will appear to be a duplicate of the header. In this case duplicate removal is critical.

Previously the tools used msgid\_comment (KDE style comments) to disambiguate text. However, with the release of Gettext 0.15, the new msgctxt disambiguation is now recommended, especially if you wish to use your files with other Gettext the tools. Many other pieces of software now also support this feature, and will probably become the best choice for almost all circumstances. It is the default in our converters.

### merge

This is the traditional Gettext approach. All messages with the same source string or English string are merged into one PO message.

```
#: file1.dtd:instruction_manual
#: file1.dtd:manual_process
msgid "Manual"
msgstr ""
```

If however the source text is blank (these are often configuration options in Mozilla) then the *merge* style will use KDE comments as used in the *msgid\_comment* style in order to create unambiguous entries that can still be used for configuration.

```
#: file1.dtd:translators_name
msgid "_: file1.dtd:translators_name\n"
msgstr ""

#: file1.dtd:translators_email
msgid "_: file1.dtd:translators_email\n"
msgstr ""
```

### msqctxt (default)

This uses the msgctxt feature of Gettext that was introduced with Gettext 0.15. Some tools might not support it 100%. This option is the default in recent releases of the Translate Toolkit.

```
#: file1.dtd:instruction_manual
msgctxt "instruction_manual"
msgid "Manual"
msgstr ""

#: file1.dtd:manual_process
msgctxt "manual_process"
msgid "Manual"
msgstr ""
```

# 1.3.31 -progress=PROGRESS

All of the programs can give visual feedback. This options allows you to select the style of that feedback.

In the examples we are converting and OpenOffice.org 2.0 sdf/gsi file into POT files using oo2po.

### none

No visual feedback, this is useful if you want to use any of the scripts as part of another script and don't want feedback to interfere with the operation.

```
$ 002po -P --progress=none en-US.sdf pot
$
```

### dots

Use visual dots to represent progress. Each dot represent a file that has been processed.

### bar (default)

Use a progress bar consisting of hashes (#) to show progress.

This is the default mode of operation, therefore this command would create the same output.

```
$ oo2po -P en-US.sdf pot
```

#### verbose

Combine the hash (#) progress bar form the bar option with the actual names of files that have been processed.

```
$ oo2po -P --progress=verbose en-US.sdf pot
processing 227 files...
so3/src.oo
dbaccess/source/ui/uno.oo
helpcontent2/source/text/shared.oo
wizards/source/formwizard.oo
sch/source/ui/dlg.oo
helpcontent2/source/text/sbasic/shared/01.oo
dbaccess/source/core/resource.oo
svtools/source/sbx.oo
dbaccess/source/ui/relationdesign.oo
scp2/source/writer.oo
filter/source/xsltdialog.oo
                                                5%
[##
                                            1
```

#### names

Prints out only the filenames without any other progress indicator. This is a good option when outputting to a log file rather than a terminal.

```
$ oo2po -P --progress=names en-US.sdf pot so3/src.oo dbaccess/source/ui/uno.oo helpcontent2/source/text/shared.oo wizards/source/formwizard.oo sch/source/ui/dlg.oo helpcontent2/source/text/sbasic/shared/01.oo dbaccess/source/core/resource.oo svtools/source/sbx.oo dbaccess/source/ui/relationdesign.oo scp2/source/writer.oo filter/source/xsltdialog.oo
```

## 1.3.32 -filteraction=ACTION

## none (default)

Take no action. Messages from failing test will appear in the output file

#### warn

Print a warning but otherwise include the message in the output file.

#### exclude-serious

Only exclude errors that are listed as serious by the convertor. All other are included.

### exclude-all

Exclude any message that fails a test.

# 1.3.33 -multifile=MULTIFILESTYLE

This options determines how the POT/PO files are spli from the source files. In many cases you have source files that generate either too many small files or one large files which you would rather see split up into smaller files.

## single

Output individual files.

## toplevel

Split the source files at the top level. Ie you see a number of top level files.

### onefiles

One large file instead of many smaller files.

# 1.3.34 -personality=TYPE

## java (default)

Create output strictly according to the specification for .properties files. This will use escaped Unicode for any non-ASCII characters. Thus the following string found in a PO file:

Will appear as follows in the output .properties file:

\u1E7D\u1E01\u1E3D\u1E7B\u1E1D

### mozilla

Mozilla has made slight adjustments to the Java .properties spec. Mozilla will accept UTF-8 encoded strings in the property file and thus does not need escaped Unicode. Thus the above string - will not be escaped. Mozilla property files are thus more useful for non-Latin languages in that they are actually readable.

Of course this style of file is only used by Mozilla and should not be used for other projects that follow the Java spec more strictly.

### skype

Skype .lang files are .properties files in UTF-16. The & is used as an accelerator (marked in the PO header).

# flex

Flex follows the Mozilla approach, a UTF-8 encoded file with no escaped unicode. We include it as its own dialect for ease of use.

## strings

Much Mac OS X and iPhone software is translated using .strings files. These are quite different from properties files and we treat them here as key value files.

The files are in UTF-16 with a few minor escaping conventions.

### 1.3.35 -accelerator=ACCELERATOR

Accelerator Marker	Used by
&	KDE Desktop and Mozilla (when using moz2po)
_	GNOME Desktop and other GTK+ based applications
~	LibreOffice and Apache OpenOffice

Converters change many different formats to PO and back again. Sometimes only one direction is supported, or conversion is done using non-PO formats. The converters follow a *general pattern of usage*, understanding that will make the converters much easier to use and understand.

• moz2po - Mozilla .properties and .dtd converter. Works with Firefox and Thunderbird

- *oo2po* OpenOffice.org SDF converter (Also works as oo2xliff).
- odf2xliff Convert OpenDocument (ODF) documents to XLIFF and vice-versa.
- prop2po Java property file (.properties) converter
- php2po PHP localisable string arrays converter.
- sub2po Converter for various subtitle files
- *txt2po* Plain text to PO converter
- po2wordfast Wordfast Translation Memory converter
- po2tmx TMX (Translation Memory Exchange) converter
- pot2po initialise PO Template files for translation
- csv2po Comma Separated Value (CSV) converter. Useful for doing translations using a spreadsheet.
- csv2tbx Create TBX (TermBase eXchange) files from Comma Separated Value (CSV) files
- html2po HTML converter
- flatxml2po Flat XML converter
- ical2po iCalendar file converter
- *ini2po* Windows INI file converter
- *json2po* JSON file converter
- web2py2po web2py translation to PO converter
- rc2po Windows Resource .rc (C++ Resource Compiler) converter
- resx2po .Net Resource (.resx) file converter
- *symb2po* Symbian-style translation to PO converter
- *tiki2po* TikiWiki language.php converter
- ts2po Qt Linguist .ts converter
- xliff2po XLIFF (XML Localisation Interchange File Format) converter
- yaml2po YAML (Yet Another Markup Language) converter

# 1.4 Tools

The PO tools allow you to manipulate and work with PO files

# 1.4.1 Quality Assurance

# poconflicts

poconflicts takes a PO file and creates an set of output PO files that contain messages that conflict. During any translation project that involves a large amount of work or a number of translators you will see message conflicts. A conflict is where the same English message has been translated differently (in some languages this may have been intentional). Conflicts occur due to different translation style or a shift in translations as the translators or project mature.

1.4. Tools 61

poconflicts allows you to quickly identify these problem messages, investigate and correct them. To merge the files back, they have to be restructured into the correct directory structure using *porestructure* in order to enable merging using *pomerge*.

## **Usage**

```
poconflicts [options] <po> <conflicts>
```

#### Where:

<po></po>	is a directory of existing PO files or an individual PO file
<conflicts></conflicts>	is a directory containing one PO file for each conflict

### Options:

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

**--progress=PROGRESS** show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in po format

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in po format

-I, --ignore-case ignore case distinctions

**-v, --invert** invert the conflicts thus extracting conflicting destination words

--accelerator=ACCELERATORS ignores the given accelerator characters when matching

# **Examples**

Here are some examples that demonstrate the usefulness of poconflict

```
poconflicts --accelerator=~ -I xhosa conflicts
```

This extracts messages from the PO files in the *xhosa* directory and places a new PO file for each identified conflict in *conflicts*. We are working with OpenOffice files and we therefore use the tilde ( $\sim$ ) as the accelerator marker (with this set  $F\sim ile$  is considered the same as  $\sim File$ ). We are also ignoring the case of the message using  $-\mathbb{I}$  (thus File is considered the same as file or FILE)

Another useful option is to look at the inverted conflicts. This will detect target words that have been used to translate different source words.

```
poconflicts --accelerator=~ -I -v xhosa conflicts
```

Now in the *conflicts* directory we will find PO files based on the Xhosa word. We can now check where a Xhosa word has been used for different source or English words. Often there is no problem but you might find cases where the same Xhosa word was used for Delete and Cancel – clearly a usability issue.

The translator makes the needed corrections to the files and then we can proceed to merge the results back into the PO files. Unchanged entries can be removed.

Now restructure the files to resemble the original directory structure using *porestructure*:

```
porestructure -i conflicts -o conflicts_tree
```

Now merge the changes back using pomerge:

```
pomerge -t xhosa -i conflicts_tree -o xhosa
```

This takes the corrected files from *conflicts\_tree* and merge them into the files in *xhosa* using the same files as templates.

## pofilter

Pofilter allows you to run a *number of checks* against your PO, XLIFF or TMX files. These checks are designed to pick up problems with capitalisation, accelerators, variables, etc. Those messages that fail any of the checks are output and marked so that you can correct them.

Use pofilter -l to get a list of available checks.

Once you have corrected the errors in your PO files you can merge the corrections into your existing translated PO files using *pomerge*.

### **Usage**

```
pofilter [options] <in> <out>
```

### Where:

<in></in>	the input file or directory which contains PO or XLIFF files	1
<out></out>	the output file or directory that contains PO or XLIFF files that fail the various tests	1

## Options:

**--version** show program's version number and exit

**-h, --help** show this help message and exit

**--manpage** output a manpage based on the help

**--progress=PROGRESS** show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in pot, po, xlf, tmx formats

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in po, pot, xlf, tmx formats

**-l, --listfilters** list filters available

**--review** include elements marked for review (default)

--noreview exclude elements marked for review
--fuzzy include elements marked fuzzy (default)

1.4. Tools 63

**--nofuzzy** exclude elements marked fuzzy

**--nonotes** don't add notes about the errors (since version 1.3)

**--autocorrect** output automatic corrections where possible rather than describing issues

--language=LANG set target language code (e.g., af-ZA) [required for spell check]. This will help to

make pofilter aware of the conventions of your language

--openoffice use the standard checks for OpenOffice translations
 --libreoffice use the standard checks for LibreOffice translations
 --mozilla use the standard checks for Mozilla translations
 --drupal use the standard checks for Drupal translations
 --gnome use the standard checks for Gnome translations
 --kde use the standard checks for KDE translations

--wx use the standard checks for wxWidgets translations – identical to –kde

--excludefilter=FILTER don't use FILTER when filtering

-t FILTER, --test=FILTER only use test FILTERs specified with this option when filtering

--notranslatefile=FILE read list of untranslatable words from FILE (must not be translated)

--musttranslatefile=FILE read list of translatable words from FILE (must be translated)

--validcharsfile=FILE read list of all valid characters from FILE (must be in UTF-8)

### **Example**

Here are some examples to demonstrate how to use pofilter:

```
pofilter --openoffice af af-check
```

Use the default settings (accelerator and variables) for OpenOffice.org. Check all PO files in *af* and output any messages that fail the check in *af-check* (create the directory if it does not already exist).

```
pofilter -t isfuzzy -t untranslated zu zu-check
```

Only run the isfuzzy and untranslated checks, this will extract all messages that are either fuzzy or untranslated.

```
pofilter --excludefilter=simplecaps --nofuzzy nso nso-check
```

Run all filters except *simplecaps*. You might want to do this if your language does not make use of capitalisation or if the test is creating too many false positives. Also only run the checks against messages that are not marked fuzzy. This is useful if you have already marked problem strings as fuzzy or you know that the fuzzy strings are bad, with this option you don't have to see the obviously wrong messages.

```
pofilter --language=fr dir dir-check
```

Tell pofilter that you are checking French translations so that it can take the conventions of the language into account (for things like punctuation, spacing, quoting, etc.) It will also disable some tests that are not meaningful for your language, like capitalisation checks for languages that don't have capital letters.

```
pofilter --excludefilter=untranslated
```

Tell pofilter not to complain about your untranslated units.

```
pofilter -1
```

List all the available checks.

### **Bugs**

There are minor bugs in the filters. Most relate to false positives, corner cases or minor changes for better fault description.

## **Descriptions of all pofilter tests**

The following are descriptions of the tests available in *pofilter*, Pootle and Virtaal with some details about what type of errors they are useful to test for and the limitations of each test.

Keep in mind that the software might point to errors which are not necessarily wrong (false positives).

Currently there are 48 tests. You can always get a list of the currently available tests by running:

```
pofilter -l
```

To see test specific to a specific targeted application or group of applications run:

```
pofilter --gnome -l
```

### Adding new tests and new language adaptations

If you have an idea for a new test or want to add target language adaptations for your language then please help us with information about your test idea and the specifics of your language.

#### **Test Classification**

Some tests are more important than others so we have classified them to help you determine which to run first.

- Critical can break a program
  - dialogsizes, escapes, newlines, nplurals, printf, pythonbraceformat, tabs, variables, xmltags
- Functional may confuse the user
  - accelerators, acronyms, blank, emails, filepaths, functions, gconf, kdecomments, long, musttranslatewords, notranslatewords, numbers, options, purepunc, sentencecount, short, spellcheck, urls, unchanged
- Cosmetic make it look better
  - brackets, doublequoting, doublespacing, doublewords, endpunc, endwhitespace, puncspacing, simplecaps, simpleplurals, startcaps, singlequoting, startpunc, startwhitespace, validchars
- Extraction useful mainly for extracting certain types of string
  - compendiumconflicts, credits, hassuggestion, isfuzzy, isreview, untranslated

1.4. Tools 65

### **Test Description**

#### accelerators

Checks whether accelerators are consistent between the two strings.

Make sure you use the --mozilla, --kde, etc options so that positive knows which type of accelerator it is looking for. The test will pick up accelerators that are missing and ones that shouldn't be there.

This check alters its default behavior in Mozilla checker for some languages so it instead checks that accelerators are not present in translation. The purpose of this is to ensure that for languages where the accelerators shouldn't be used the accelerators are not present in the translations. This is common for Indic languages.

### acronyms

Checks that acronyms that appear are unchanged.

If an acronym appears in the original this test will check that it appears in the translation. Translating acronyms is a language decision but many languages leave them unchanged. In that case this test is useful for tracking down translations of the acronym and correcting them.

#### blank

Checks whether a translation is totally blank.

This will check to see if a translation has inadvertently been translated as blank i.e. as spaces. This is different from untranslated which is completely empty. This test is useful in that if something is translated as "" it will appear to most tools as if it is translated.

### brackets

Checks that the number of brackets in both strings match.

If ([{ or }]) appear in the original this will check that the same number appear in the translation.

### compendiumconflicts

Checks for Gettext compendium conflicts (#-#-#-#-#).

When you use msgcat to create a PO compendium it will insert #-#-#-# into entries that are not consistent. If the compendium is used later in a message merge then these conflicts will appear in your translations. This test quickly extracts those for correction.

### credits

Checks for messages containing translation credits instead of normal translations.

Some projects have consistent ways of giving credit to translators by having a unit or two where translators can fill in their name and possibly their contact details. This test allows you to find these units easily to check that they are completed correctly and also disables other tests that might incorrectly get triggered for these units (such as urls, emails, etc.)

# dialogsizes

Checks that dialog sizes are not translated.

This is a Mozilla specific test. Mozilla uses a language called XUL to define dialogues and screens. This can make use of CSS to specify properties of the dialogue. These properties include things such as the width and height of the box. The size might need to be changed if the dialogue size changes due to longer translations. Thus translators can change these settings. But you are only meant to change the number not translate the words 'width' or 'height'. This check capture instances where these are translated. It will also catch other types of errors in these units.

# doublequoting

Checks whether doublequoting is consistent between the two strings.

Checks on double quotes " to ensure that you have the same number in both the original and the translated string. This tests takes into account that several languages use different quoting characters, and will test for them instead.

# doublespacing

Checks for bad double-spaces by comparing to original.

This will identify if you have [space][space] in when you don't have it in the original or it appears in the original but not in your translation. Some of these are spurious and how you correct them depends on the conventions of your language.

### doublewords

Checks for repeated words in the translation.

Words that have been repeated in a translation will be highlighted with this test e.g. "the the", "a a". These are generally typos that need correcting. Some languages may have valid repeated words in their structure, in that case either ignore those instances or switch this test off using the --excludefilters option.

### emails

Checks to see that emails are not translated.

Generally you should not be translating email addresses. This check will look to see that email addresses e.g. info@example.com are not translated. In some cases of course you should translate the address but generally you shouldn't.

# endpunc

Checks whether punctuation at the end of the strings match.

This will ensure that the ending of your translation has the same punctuation as the original. E.g. if it ends in :[space] then so should yours. It is useful for ensuring that you have ellipses [...] in all your translations, not simply three separate full-stops. You may pick up some errors in the original: feel free to keep your translation and notify the programmers. In some languages, characters such as ?! are always preceded by a space e.g. [space]? — do what your language customs dictate. Other false positives you will notice are, for example, if through changes in word-order you add "), etc. at the end of the sentence. Do not change these: your language word-order takes precedence.

1.4. Tools 67

It must be noted that if you are tempted to leave out [full-stop] or [colon] or add [full-stop] to a sentence, that often these have been done for a reason, e.g. a list where fullstops make it look cluttered. So, initially match them with the English, and make changes once the program is being used.

This check is aware of several language conventions for punctuation characters, such as the custom question marks for Greek and Arabic, Devenagari Danda, full-width punctuation for CJK languages, etc. Support for your language can be added easily if it is not there yet.

### endwhitespace

Checks whether whitespace at the end of the strings matches.

Operates the same as endpunc but is only concerned with whitespace. This filter is particularly useful for those strings which will evidently be followed by another string in the program, e.g. [Password: ] or [Enter your username: ]. The whitespace is an inherent part of the string. This filter makes sure you don't miss those important but otherwise invisible spaces!

If your language uses full-width punctuation (like Chinese), the visual spacing in the character might be enough without an added extra space.

## escapes

Checks whether escaping is consistent between the two strings.

Checks escapes such as \n \uNNNN to ensure that if they exist in the original string you also have them in the translation.

### filepaths

Checks that file paths have not been translated.

Checks that paths such as /home/user1 have not been translated. Generally you do not translate a file-path, unless it is being used as an example, e.g. [your\_user\_name/path/to/filename.conf].

### **functions**

Checks to see that function names are not translated.

Checks that function names e.g. rgb() or getEntity.Name() are not translated.

# gconf

Checks if we have any gconf config settings translated.

Gconf settings should not be translated so this check checks that gconf settings such as "name" or "modification\_date" are not translated in the translation. It allows you to change the surrounding quotes but will ensure that the setting values remain untranslated.

# hassuggestion

Checks if there is at least one suggested translation for this unit.

If a message has a suggestion (an alternate translation stored in alt-trans units in XLIFF and .pending files in PO) then these will be extracted. This is used by Pootle and is probably only useful in pofilter when using XLIFF files.

# isfuzzy

Checks if the po element has been marked fuzzy.

If a message is marked fuzzy in the PO file then it is extracted. Note this is different from --fuzzy and --nofuzzy options which specify whether tests should be performed against messages marked fuzzy.

### isreview

Checks if the po element has been marked for review.

If you have made use of the 'review' flags in your translations:

```
# (review) reason for review
# (pofilter) testname: explanation for translator
```

Then if a message is marked for review in the PO file it will be extracted. Note this is different from --review and --noreview options which specify whether tests should be performed against messages already marked as under review.

#### kdecomments

Checks to ensure that no KDE style comments appear in the translation.

KDE style translator comments appear in PO files as "\_: comment\n". New translators often translate the comment. This test tries to identify instances where the comment has been translated.

### long

Checks whether a translation is much longer than the original string.

This is most useful in the special case where the translation is multiple characters long while the source text is only 1 character long. Otherwise, we use a general ratio that will catch very big differences but is set conservatively to limit the number of false positives.

### musttranslatewords

Checks that words configured as definitely translatable don't appear in the translation.

If for instance in your language you decide that you must translate 'OK' then this test will flag any occurrences of 'OK' in the translation if it appeared in the source string. You must specify a file containing all of the *must translate* words using <code>--musttranslatefile</code>.

#### newlines

Checks whether newlines are consistent between the two strings.

Counts the number of \n newlines (and variants such as \r\n) and reports and error if they differ.

### nplurals

Checks for the correct number of noun forms for plural translations.

This uses the plural information in the language module of the toolkit. This is the same as the Gettext nplural value. It will check that the number of plurals required is the same as the number supplied in your translation.

#### notranslatewords

Checks that words configured as untranslatable appear in the translation too.

Many brand names should not be translated, this test allows you to easily make sure that words like: Word, Excel, Impress, Calc, etc. are not translated. You must specify a file containing all of the *no translate* words using --notranslatefile.

#### numbers

Checks whether numbers of various forms are consistent between the two strings.

You will see some errors where you have either written the number in full or converted it to the digit in your translation. Also changes in order will trigger this error.

Some languages don't use latin numbers but instead use different numbers. This check will take that into account.

### options

Checks that command line options are not translated.

In messages that contain command line options, such as --help, this test will check that these remain untranslated. These could be translated in the future if programs can create a mechanism to allow this, but currently they are not translated. If the options has a parameter, e.g. --file=FILE, then the test will check that the parameter has been translated.

### printf

Checks whether printf format strings match.

If the printf formatting variables are not identical, then this will indicate an error. Printf statements are used by programs to format output in a human readable form (they are place holders for variable data). They allow you to specify lengths of string variables, string padding, number padding, precision, etc. Generally they will look like this: %d, %5.2f, %100s, etc. The test can also manage variables-reordering using the %1\$s syntax. The variables' type and details following data are tested to ensure that they are strictly identical, but they may be reordered.

#### See also:

pythonbraceformat

See also:

### printf Format String

# puncspacing

Checks for bad spacing after punctuation.

In the case of [full-stop][space] in the original, this test checks that your translation does not remove the space. It checks also for [comma], [colon], etc.

Some languages don't use spaces after common punctuation marks, especially where full-width punctuation marks are used. This check will take that into account.

# purepunc

Checks that strings that are purely punctuation are not changed.

This extracts strings like "+" or "-" as these usually should not be changed.

### pythonbraceformat

Checks whether Python brace format strings match.

Python supports both a variant of the *printf* formatting system, and its own formatting language which uses placeholders enclosed in braces. The placeholders can be named, numbered, or anonymous; the former two are filled in from positional args, the latter from keyword arguments. Example:

```
'the {} {0} hungry {insect}'.format('very', insect='caterpiller')
# --> 'the very very hungry caterpiller'
```

The pythonbraceformat filter checks for the following problems:

- named placeholders that are present in the original, but missing in the translation, and vice versa.
- originals and translations that require different numbers of positional args.

When the translation has variables not in the original, this can lead to program crashes. The translation not using all variables the original uses is safe. Nonetheless, this filter triggers in both cases.

### See also:

PEP 3101 – Advanced String Formatting

#### sentencecount

Checks that the number of sentences in both strings match.

Adds the number of sentences to see that the sentence count is the same between the original and translated string. You may not always want to use this test, if you find you often need to reformat your translation, because the original is badly-expressed, or because the structure of your language works better that way. Do what works best for your language: it's the meaning of the original you want to convey, not the exact way it was written in the English.

#### short

Checks whether a translation is much shorter than the original string.

This is most useful in the special case where the translation is 1 characters long while the source text is multiple characters long. Otherwise, we use a general ratio that will catch very big differences but is set conservatively to limit the number of false positives.

### simplecaps

Checks the capitalisation of two strings isn't wildly different.

This will pick up many false positives, so don't be a slave to it. It is useful for identifying translations that don't start with a capital letter (upper-case letter) when they should, or those that do when they shouldn't. It will also highlight sentences that have extra capitals; depending on the capitalisation convention of your language, you might want to change these to Title Case, or change them all to normal sentence case.

# simpleplurals

Checks for English style plural(s) for you to review.

This test will extract any message that contains words with a final "(s)" in the source text. You can then inspect the message, to check that the correct plural form has been used for your language. In some languages, plurals are made by adding text at the beginning of words, making the English style messy. In this case, they often revert to the plural form. This test allows an editor to check that the plurals used are correct. Be aware that this test may create a number of false positives.

For languages with no plural forms (only one noun form) this test will simply test that nothing like "(s)" was used in the translation.

### singlequoting

Checks whether singlequoting is consistent between the two strings.

The same as doublequoting but checks for the 'character. Because this is used in contractions like it's and in possessive forms like user's, this test can output spurious errors if your language doesn't use such forms. If a quote appears at the end of a sentence in the translation, i.e. '., this might not be detected properly by the check.

#### spellcheck

Checks for words that don't pass a spell-check.

This test will check for misspelled words in your translation. The test first checks for misspelled words in the original (usually English) text, and adds those to an exclusion list. The advantage of this exclusion is that many words that are specific to the application will not raise errors e.g. program names, brand names, function names.

The checker works with PyEnchant. You need to have PyEnchant installed as well as a dictionary for your language (for example, one of the Hunspell or aspell dictionaries). This test will only work if you have specified the --language option.

The pofilter error that is created, lists the misspelled word, plus suggestions returned from the spell checker. That makes it easy for you to identify the word and select a replacement.

### startcaps

Checks that the message starts with the correct capitalisation.

After stripping whitespace and common punctuation characters, it then checks to see that the first remaining character is correctly capitalised. So, if the sentence starts with an upper-case letter, and the translation does not, an error is produced.

This check is entirely disabled for many languages that don't make a distinction between upper and lower case. Contact us if this is not yet disabled for your language.

### startpunc

Checks whether punctuation at the beginning of the strings match.

Operates as endpunc but you will probably see fewer errors.

### startwhitespace

Checks whether whitespace at the beginning of the strings matches.

As in endwhitespace but you will see fewer errors.

#### tabs

Checks whether tabs are consistent between the two strings.

Counts the number of \t tab markers and reports an error if they differ.

### unchanged

Checks whether a translation is basically identical to the original string.

This checks to see if the translation isn't just a copy of the English original. Sometimes, this is what you want, but other times you will detect words that should have been translated.

#### untranslated

Checks whether a string has been translated at all.

This check is really only useful if you want to extract untranslated strings so that they can be translated independently of the main work.

#### urls

Checks to see that URLs are not translated.

This checks only basic URLs (http, ftp, mailto etc.) not all URIs (e.g. afp, smb, file). Generally, you don't want to translate URLs, unless they are example URLs (http://your\_server.com/filename.html). If the URL is for configuration information, then you need to query the developers about placing configuration information in PO files. It shouldn't really be there, unless it is very clearly marked: such information should go into a configuration file.

#### validchars

Checks that only characters specified as valid appear in the translation.

Often during character conversion to and from UTF-8 you get some strange characters appearing in your translation. This test presents a simple way to try and identify such errors.

This test will only run of you specify the --validcharsfile command line option. This file contains all the characters that are valid in your language. You must use UTF-8 encoding for the characters in the file.

If the test finds any characters not in your valid characters file then the test will print the character together with its Unicode value (e.g. 002B).

#### variables

Checks whether variables of various forms are consistent between the two strings.

This checks to make sure that variables that appear in the original also appear in the translation. Make sure you use the --kde, --openoffice, etc flags as these define what variables will be searched for. It does not at the moment cope with variables that use the reordering syntax of Gettext PO files.

#### **xmltags**

Checks that XML/HTML tags have not been translated.

This check finds the number of tags in the source string and checks that the same number are in the translation. If the counts don't match then either the tag is missing or it was mistakenly translated by the translator, both of which are errors.

The check ignores tags or things that look like tags that cover the whole string e.g. "<Error>" but will produce false positives for things like "An <Error> occurred" as here "Error" should be translated. It also will allow translation of the alt attribute in e.g. <img src=bob.png alt="Image description"> or similar translatable attributes in OpenOffice.org help files.

#### pogrep

The pogrep tool extracts messages that match a regular expression into a new set of PO files that can be examined, edited and corrected. These corrections can then be merged using *pomerge*.

# **Usage**

```
pogrep [options] <in> <out>
```

#### Where:

<in>/<out > n and out are either directories or files. Out will contain PO/XLIFF files with only those messages that match the regular expression that was you searched for.

### Options:

**--version** show program's version number and exit

**-h, --help** show this help message and exit

```
--manpage output a manpage based on the help
```

- --progress=PROGRESS show progress as: dots, none, bar, names, verbose
- --errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback
- -i INPUT, --input=INPUT read from INPUT in gmo, mo, po, pot, tmx, xlf, xlff, xliff formats
- -x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths
- -o OUTPUT, --output=OUTPUT write to OUTPUT in gmo, mo, po, pot, tmx, xlf, xlff, xliff formats
- --search=SEARCHPARTS searches the given parts (source, target, notes, locations)
- **-I, --ignore-case** ignore case distinctions
- **-e, --regexp** use regular expression matching
- -v, --invert-match select non-matching lines
- --accelerator=ACCELERATORS ignores the given accelerator characters when matching
- -k, --keep-translations always extract units with translations

# **Example**

```
pogrep --accelerator="_" --search msgid -I -e "software|hardware" only-zu only-zu-

→check
```

Search for the words "software" or "hardware" in the msgid field. Ignore case (-I) and treat the underscore (\_) character as an accelerator key. Search through all PO files in the directory "only-zu" and place any matches in PO files in the directory "only-zu-check". This would be useful to run if you know that the word for software and hardware has been changed during the course of translation and you want to check and correct all these instances.

```
pogrep --search=msgid -e \ '^w+(\s+\w+) \{0,3\}' -i templates -o short-words
```

Find all messages in the *templates* directory that have between 1 and 4 words and place them in *short-words*. Use this if you want to see quick results by translating messages that are most likely menu entries or dialogue labels.

```
pogrep --search=msgstr -I -e "Ifayile" zu zu-check
```

Search all translations for the occurrence of *Ifayile*. You would use this to check if words have been used correctly. Useful if you find problematic use of the same word for different concepts. You can use *pocompendium* to find these conflicts.

### **Notes**

### **Unicode normalization**

pogrep will normalize Unicode strings. This allows you to search for strings that contain the same character but that are using precomposed Unicode characters or which are composed using another composition recipe. While an individual user will in all likelihood only compose characters in one way, normalization ensures that data created in a team setting can be shared.

### **Further reading**

Here is a blog post explaining how pogrep can be used to do more targeted localisation of GNOME: http://translate.org.za/blogs/friedel/en/content/better-lies-about-gnome-localisation

### pomerge

Pomerge will merge corrected PO, XLIFF, or TMX files (or snippets) into your existing PO, XLIFF, TMX files. Usually you would extract errors using *pofilter*, make corrections to these PO (or XLIFF, TMX) snippets then merge them back using pomerge. You could also use *pogrep* to extract a number of messages matching a certain string, make corrections then merge the correction back using pomerge.

It is probably best to run pomerge against files stored in some kind of version control system so that you can monitor what changes were made.

Pomerge will also attempt to make as small a change as possible to the text, making it easier to see the changes using your version control system.

# **Usage**

```
pomerge [options] [-t <template>] -i <input> -o <output>
```

#### Where:

<tem-< th=""><th>is a set of reference PO, XLIFF, TMX files, either the originals or a set of POT files</th></tem-<>	is a set of reference PO, XLIFF, TMX files, either the originals or a set of POT files
plate>	
<input/>	contains the corrected files that are to override content in <output></output>
<output></output>	contains the files whose content will be overridden by <input/> . This can be the same directory as
	<template></template>

# Options:

**--version** show program's version number and exit

**-h, --help** show this help message and exit

**--manpage** output a manpage based on the help

- --progress=PROGRESS show progress as: dots, none, bar, names, verbose
- --errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback
- -i INPUT, --input=INPUT read from INPUT in po, pot, xlf formats
- -x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths
- -o OUTPUT, --output=OUTPUT write to OUTPUT in po, pot, xlf formats
- -t TEMPLATE, --template=TEMPLATE read from TEMPLATE in po, pot, xlf formats
- **-S**, **--timestamp** skip conversion if the output file has newer timestamp
- --mergeblanks=MERGEBLANKS whether to overwrite existing translations with blank translations (yes/no). Default is yes.
- --mergefuzzy=MERGEFUZZY whether to overwrite existing translations with fuzzy translations (yes/no). Default is yes.

--mergecomments=MERGECOMMENTS whether to merge comments as well as translations (yes/no). Default is yes.

### **Examples**

These examples show pomerge in action.

```
pomerge -t af -i af-check -o af
```

Take corrections from *af-check* merge them with the templates in *af* and output into *af*. Thus merge af-check and override entries found in *af*. Do this only if you are using a version control system so that you can check what changes pomerge made or if you have complete and utter confidence in this tool.

```
pomerge --mergeblanks=yes -t af -i af-check -o af-new
```

Merge the corrections from *af-check* with templates in *af* and output to *af-new*. If an entry is blank in *af-check* then make it blank in the output in *af-new*.

#### Issues

- Seems to have trouble merging KDE style comments back. (Probably not relevant with newest versions any more.)
- Only files found in the input directory will be copied to the output. The template directory is not searched for extra files to copy to the output. Therefore it is always best to have your input directory in version control, and use the same directory as output. This will allow you to use the diff function of the version control system to double check changes made, with all the files of the input still present.

#### porestructure

porestructure takes the PO files output by *poconflicts* (a flat structure), and recreates the directory structure according to the poconflict location comments found in each PO message. After being restructured, the messages in the resulting directory structure can be merged back using *pomerge*.

Since poconflicts adds conflicting messages, from many different PO files, into a single PO file, the original structure of the files and directories are lost and the new PO files are output to a single directory. The original structure information is left in "(pofilter)" comments for each PO element.

### **Usage**

```
porestructure [options] <conflicts> <po>
```

#### Where:

<conflicts></conflicts>	is a directory containing one the corrected output from poconflict
<po></po>	is an output directory to write the restructured files to

# Options:

--version show program's version number and exit-h, --help show this help message and exit

- **--manpage** output a manpage based on the help
- --progress=PROGRESS show progress as: dots, none, bar, names, verbose
- --errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback
- -i INPUT, --input=INPUT read from INPUT in po format
- -x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths
- -o OUTPUT, --output=OUTPUT write to OUTPUT in po format

### **Examples**

The documentation for poconflicts has *Examples* for the complete process using poconflict, porestructure, and pomerge.

# junitmsgfmt

New in version 1.7.

Run msgfmt and provide JUnit type output for use in continuous integration systems like Hudson and Jenkins.

# **Usage**

```
junitmsgfmt po/*.po > msgfmt_junit.xml
```

These tools are especially useful for measuring and improving translation quality.

- poconflicts extract messages that have conflicting translation
- pofilter filter PO files to find common errors using a number of tests
- pogrep find strings in your PO files
- pomerge merge file extracted using pofilter back into the original files
- porestructure restructures PO files according to poconflict directives
- junitmsgfmt run msgfmt and provide JUnit type output for use in continuous integration systems like Hudson and Jenkins

# 1.4.2 Other tools

#### tmserver

tmserver is a Translation Memory service that can be queried via HTTP using a simple REST like URL/http and data is exchanged between server and client encoded in JSON.

**Note:** If you are searching for an enterprise Translation Memory server then rather use amaGama.

# **Usage**

#### Where:

TMDB-	is the SQLite database file containing translation memory data, if not specified a new temporary	
FILE	database is created	
TMFILE	E is a translation file (po, xliff, etc.) that should be imported into the database (mostly useful when no	
	tmdb file is specified).	

### Options:

- **-h, --help** show this help message and exit
- -d TMDBFILE, --tmdb=TMDBFILE translation memory database file
- -f TMFILES, --import-translation-file=TMFILES translation file to import into the database
- -t TARGET\_LANG, --import-target-lang=TARGET\_LANG target language of translation files
- -s SOURCE\_LANG, --import-source-lang=SOURCE\_LANG source language of translation files
- **-b BIND, --bind=BIND** address to bind server to (default: localhost)
- -p PORT, --port=PORT port to listen on (default: 8888)
- --max-candidates=MAX\_CANDIDATES Maximum number of candidates
- --min-similarity=MIN\_SIMILARITY minimum similarity
- --max-length=MAX\_LENGTH Maxmimum string length
- **--debug** enable debugging features

### **Testing**

easiest way to run the server for testing is to pass it a large translation file (maybe generated by *pocompendium*) to create a tmdb database on the fly.

```
tmserver -b localhost -p 8080 -f compendium.po -s en_US -t ar
```

The server can be queried using a webbrowser. the url would be:

```
http://HOST:PORT/tmserver/SOURCE_LANG/TARGET_LANG/unit/STRING
```

So to see suggestions for "open file" try the url http://localhost:8080/tmserver/en\_US/ar/unit/open+file

#### poterminology

poterminology takes Gettext PO/POT files and extracts potential terminology.

This is useful as a first step before translating a new project (or an existing project into a new target language) as it allows you to define key terminology for consistency in translations. The resulting terminology PO files can be used by Pootle to provide suggestions while translating.

Generally, all the input files should have the same source language, and either be POT files (with no translations) or PO files with translations to the same target language.

The more separate PO files you use to generate terminology, the better your results will be, but poterminology can be used with just a single input file.

Read more about terminology extraction

### **Usage**

```
poterminology [options] <input> <terminology>
```

#### Where:

<input/>	translations to be examined for terminology
<terminology></terminology>	extracted potential terminology

# Options:

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

- **--progress=PROGRESS** show progress as: dots, none, bar, names, verbose
- --errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback
- -i INPUT, --input=INPUT read from INPUT in pot, po formats
- -x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths
- -o OUTPUT, --output=OUTPUT write to OUTPUT in po, pot formats
- -u UPDATEFILE, --update=UPDATEFILE update terminology in UPDATEFILE
- **-S STOPFILE**, **--stopword-list=STOPFILE** read stopword (term exclusion) list from STOPFILE (default site-packages/translate/share/stoplist-en)
- **-F, --fold-titlecase** fold "Title Case" to lowercase (default)
- -C, --preserve-case preserve all uppercase/lowercase
- **-I, --ignore-case** make all terms lowercase
- --accelerator=ACCELERATORS ignore the given accelerator characters when matching (accelerator characters probably require quoting)
- **-t LENGTH, --term-words=LENGTH** generate terms of up to LENGTH words (default 3)
- --nonstop-needed=MIN omit terms with less than MIN nonstop words (default 1)
- **--inputs-needed=MIN** omit terms appearing in less than MIN input files (default 2, or 1 if only one input file)
- **--fullmsg-needed=MIN** omit full message terms appearing in less than MIN different messages (default 1)
- **--substr-needed=MIN** omit substring-only terms appearing in less than MIN different messages (default 2)

```
    --locs-needed=MIN omit terms appearing in less than MIN different original program locations (default 2)
    --sort=ORDER output sort order(s): frequency, dictionary, length (default is all orders in the above priority)
```

--source-language=LANG the source language code (default 'en')

**-v, --invert** invert the source and target languages for terminology

### **Examples**

You want to generate a terminology file for Pootle that will be used to provide suggestions for translating Pootle itself:

```
poterminology Pootle/po/pootle/templates/*.pot .
```

This results in a ./pootle-terminology.pot output file with 23 terms (from "file" to "does not exist") — without any translations.

The default output file can be added to a Pootle project to provide terminology matching suggestions for that project; alternately a special Terminology project can be used and it will provide terminology suggestions for all projects that do not have a pootle-terminology.po file.

Generating a terminology file containing automatically extracted translations is possible as well, by using PO files with translations for the input files:

```
poterminology Pootle/po/pootle/fi/*.po --output fi/pootle-terminology.po --sort_ --dictionary
```

Using PO files with Finnish translations, you get an output file that contains the same 23 terms, with translations of eight terms – one ("login") is fuzzy due to slightly different translations in jToolkit and Pootle. The file is sorted in alphabetical order (by source term, not translated term), which can be useful when comparing different terminology files.

Even though there is no translation of Pootle into Kinyarwanda, you can use the Gnome UI terminology PO file as a source for translations; in order to extract only the terms common to jToolkit and Pootle this command includes the POT output from the first step above (which is redundant) and require terms to appear in three different input sources:

```
poterminology Pootle/po/pootle/templates/*.pot pootle-terminology.pot \
  Pootle/po/terminology/rw/gnome/rw.po --inputs-needed=3 -o terminology/rw.po
```

Of the 23 terms, 16 have Kinyarwanda translations extracted from the Gnome UI terminology.

For a language like Spanish, with both Pootle translations and Gnome terminology available, 18 translations (2 fuzzy) are generated by the following command, which initializes the terminology file from the POT output from the first step, and then uses --update to specify that the pootle-es.po file is to be used both for input and output:

```
cp pootle-terminology.pot glossary-es.po
poterminology --inputs=3 --update glossary-es.po \
  Pootle/po/pootle/es/*.po Pootle/po/terminology/es/gnome/es.po
```

### Reduced terminology glossaries

If you want to generate a terminology file containing only single words, not phrases, you can use -t/--term-words to control this. If your input files are very large and/or you have a lot of input files, and you are finding that poterminology is taking too much time and memory to run, reducing the phrase size from the default value of 3 can be helpful.

For example, running poterminology on the subversion trunk with the default phrase size can take quite some time and may not even complete on a small-memory system, but with --term-words=1 the initial number of terms is reduced by half, and the thresholding process can complete:

```
poterminology --progress=none -t 1 translate

1297 terms from 64039 units in 216 files
254 terms after thresholding
254 terms after subphrase reduction
```

The first line of output indicates the number of input files and translation units (messages), with the number of unique terms present after removing C and Python format specifiers (e.g. %d), XML/HTML <elements> and &entities; and performing stoplist elimination.

The second line gives the number of terms remaining after applying threshold filtering (discussed in more detail below) to eliminate terms that are not sufficiently "common" in the input files.

The third line gives the number of terms remaining after eliminating subphrases that did not occur independently. In this case, since the term-words limit is 1, there are no subphrases and so the number is the same as on the second line.

However, in the first example above (generating terminology for Pootle itself), the term "not exist" passes the stoplist and threshold filters, but all occurrences of this term also contained the term "does not exist" which also passes the stoplist and threshold filters. Given this duplication, the shorter phrase is eliminated in favor of the longer one, resulting in 23 terms (out of 25 that pass the threshold filters).

### Reducing output terminology with thresholding options

Depending on the size and number of the source files, and the desired scope of the output terminology file, there are several thresholding filters that can be adjusted to allow fewer or more terms in the output file. We have seen above how one (--inputs-needed) can be used to require that terms be present in multiple input files, but there are also other thresholds that can be adjusted to control the size of the output terminology file.

### -inputs-needed

This is the most flexible and powerful thresholding control. The default value is 2, unless only one input file (not counting an --update argument) is provided, in which case the threshold is 1 to avoid filtering out all terms and generating an empty output terminology file.

By copying input files and providing them multiple times as inputs, you can even achieve "weighted" thresholding, so that for example, all terms in one original input file will pass thresholding, while other files may be filtered. A simple version of this technique was used above to incorporate translations from the Gnome terminology PO files without having it affect the terms that passed the threshold filters.

# -locs-needed

Rather than requiring that a term appear in multiple input PO or POT files, this requires that it have been present in multiple source code files, as evidenced by location comments in the PO/POT sources.

This threshold can be helpful in eliminating over-specialized terminology that you don't want when multiple PO/POT files are generated from the same sources (via included header or library files).

Note that some PO/POT files have function names rather than source file names in the location comments; in this case the threshold will be on multiple functions, which may need to be set higher to be effective.

Not all PO/POT files contain proper location comments. If your input files don't have (good) location comments and the output terminology file is reduced to zero or very few entries by thresholding, you may need to override the default value for this threshold and set it to 0, which disables this check.

The setting of the --locs-needed comment has another effect, which is that location comments in the output terminology file will be limited to twice that number; a location comment indicating the number of additional locations not specified will be added instead of the omitted locations.

# -fullmsg-needed & -substr-needed

These two thresholds specify the number of different translation units (messages) in which a term must appear; they both work in the same way, but the first one applies to terms which appear as complete translation units in one or more of the source files (full message terms), and the second one to all other terms (substring terms). Note that translations are extracted only for full message terms; poterminology cannot identify the corresponding substring in a translation.

If you are working with a single input file without useful location comments, increasing these thresholds may be the only way to effectively reduce the output terminology. Generally, you should increase the --substr-needed threshold first, as the full message terms are more likely to be useful terminology.

# Stop word files

Much of the power of poterminology in generating useful terminology files is due to the default stop word file that it uses. This file contains words and regular expressions that poterminology will ignore when generating terms, so that the output terminology doesn't have tons of useless entries like "the 16" or "Z".

In most cases, the default stop word list will work well, but you may want to replace it with your own version, or possibly just supplement or override certain entries. The default *poterminology stopword file* contains comments that describe the syntax and operation of these files.

If you want to completely replace the stopword list (for example, if your source language is French rather than English) you could do it with a command like this:

```
poterminology --stopword-list=stoplist-fr logiciel/ -o glossaire.po
```

If you merely want to modify the standard stopword list with your own additions and overrides, you must explicitly specify the default list first:

```
poterminology -S /usr/lib/python2.5/site-packages/translate/share/stoplist-en \
   -S my-stoplist po/ -o terminology.po
```

You can use poterminology --help to see the default stopword list pathname, which may differ from the one shown above.

Note that if you are using multiple stopword list files, as in the above, they will all be subject to the same case mapping (fold "Title Case" to lower case by default) – if you specify a different case mapping in the second file it will override the mapping for all the stopword list files.

#### Issues

When using poterminology on Windows systems, file globbing for input is not supported (unless you have a version of Python built with cygwin, which is not common). On Windows, a command like poterminology -o test.po podir/\\*.po will fail with an error "No such file or directory: 'podir\\*.po" instead of expanding the podir/\*.po glob expression. (This problem affects all Translate Toolkit command-line tools, not just poterminology.) You can work around this problem by making sure that the directory does not contain any files (or subdirectories) that you do

not want to use for input, and just giving the directory name as the argument, e.g. poterminology -o test.po podir for the case above.

When using terminology files generated by poterminology as input, a plethora of translator comments marked with (poterminology) may be generated, with the number of these increasing on each iteration. You may wish to run *pocommentclean* (or a slightly modified version of it which only removes (poterminology) comments) on the input and/or output files, especially since translator comments are displayed as tooltips by Pootle (thankfully, they are truncated at a few dozen characters).

Default threshold settings may eliminate all output terms; in this case, poterminology should suggest threshold option settings that would allow output to be generated (this enhancement is tracked as issue 582).

While poterminology ignores XML/HTML entities and elements and %-style format strings (for C and Python), it does not ignore all types of "variables" that may occur, particularly in OpenOffice.org, Mozilla, or Gnome localization files. These other types should be ignored as well (this enhancement is tracked as issue 598).

Terms containing only words that are ignored individually, but not excluded from phrases (e.g. "you are you") may be generated by poterminology, but aren't generally useful. Adding a new threshold option --nonstop-needed could allow these to be suppressed (this enhancement is tracked as issue 1102).

Pootle ignores parenthetical comments in source text when performing terminology matching; this allows for terms like "scan (verb)" and "scan (noun)" to both be provided as suggestions for a message containing "scan." poterminology does not provide any special handling for these, but it could use them to provide better handling of different translations for a single term. This would be an improvement over the current approach, which marks the term fuzzy and includes all variants, with location information in {} braces in the automatically extracted translation.

Currently, message context information (PO msgctxt) is not used in any way; this could provide an additional source of information for distinguishing variants of the same term.

A single execution of poterminology can only perform automatic translation extraction for a single target language – having the ability to handle all target languages in one run would allow a single command to generate all terminology for an entire project. Additionally, this could provide even more information for identifying variant terms by comparing the number of target languages that have variant translations.

# On single files

If poterminology yields 0 terms from single files, try the following:

```
poterminology --locs-needed=0 --inputs-needed=0 --substr-needed=5 -i yourfile.po -o_

→yourfile_term.po
```

... where "substr-needed" is the number of times a term should occur to be considered.

# Stopword file format

New in version 1.2.

The default stopword file for *poterminology* describes the syntax of these files and provides a good default for most applications using English source text. You can find the location of the default stopword file by looking at the output of poterminology —help, or using the following command:

```
poterminology --manpage | sed -n '/STOPFILE/s/.*(\((.*\))).*/\1/p'
```

### **Overview**

The basic syntax of this file is line-oriented, with the first character of each line determining its function. The order of the lines is generally not significant (with one exception noted below), and the selection of function characters was made so that an ASCII sort of the file would leave it in a generally logical order (except for comment lines).

Apart from comment lines (which begin with '#') and empty lines (which are also ignored), there are three general types of lines, which may appear in any order:

- · case mapping specifiers
- · stoplist regular expressions
- · stoplist words

### Case mapping specifiers

A line beginning with a '!' specifies upper-/lower-case mapping for words or phrases before comparison with this stoplist (no mapping is applied to the words or regular expressions in this file, only to the source messages). The second character on this line must be one of the following:

- C no uppercase / lowercase mapping is performed
- F 'Title Case" words / terms are folded to lower case (default)
- I all words are mapped to lowercase

These correspond to the equivalent --preserve-case / --fold-titlecase / --ignore-case options to poterminology, but are completely independent and only apply to stoplist matching. You can run poterminology with -I to map all terms to lowercase, and if the case mapping specifier in the stopword file is '!C' a stoplist with "pootle" in it will not prevent a term containing "Pootle" from passing the stoplist (and then being mapped to "pootle").

There should only be one case mapping specifier in a stoplist file; if more than one are present, the last one will take precedence over the others, and its mapping will apply to all entries. If multiple stoplist files are used, the last case mapping specifier processed will apply to all entries **in all files**.

# Stoplist regular expressions

Lines beginning with a '/' are regular expression patterns – any word that matches will be ignored by itself, and any phrase containing it will be excluded as well. The regular expression consists of all characters on the line following the initial '/' – these are extended regular expressions, so grouping, alternation, and such are available.

Regular expression patterns are only checked if the word itself does not appear in the stoplist file as a word entry. The regular expression patterns are always applied to individual words, not phrases, and must match the entire word (i.e. they are anchored both at the start and end).

Use regular expressions sparingly, as evaluating them for every word in the source files can be expensive. In addition to stoplist regular expressions, poterminology has precompiled patterns for C and Python format specifiers (e.g. %d) and XML/HTML <elements> and &entities; – these are removed before stoplist processing and it is not possible to override this.

### **Stoplist words**

All other lines should begin with one of the following characters, which indicate whether the word should be **ignored** (as a word alone), **disregarded** in a phrase (i.e. a phrase containing it is allowed, and the word does not count against the --term-words length limit), or any phrase containing it should be **excluded**.

- + allow word alone, allow phrases containing it
- : allow word alone, disregarded (for --term-word-length) inside phrase
- < allow word alone, but exclude any phrase containing it
- = ignore word alone, but allow phrases containing it
- > ignore word alone, disregarded (for --term-word-length) inside phrase
- @ ignore word alone, and exclude any phrase containing it

Generally '+' is only needed for exceptions to regular expression patterns, but it may also be used to override an entry in a previous stoplist if you are using multiple stoplists.

Note that if a word appears multiple times in a stoplist file with different function characters preceding it, the *last* entry will take precedence over the others. This is the only exception to the general rule that order is not important in stopword files.

# Default file example

```
\mbox{\#} apply title-case folding to words before comparing with this stoplist \mbox{!} F
```

The fold-titlecase setting is the default, even if it were not explicitly specified. This allows capitalized words at the start of a sentence (e.g. "Who") to match a stopword "who" but allows acronyms like WHO (World Health Organization) to be included in the terminology. If you are using poterminology with source files that contain large amounts of ALL UPPERCASE TEXT you may find the ignore-case setting to be preferable.

```
# override regex match below for phrases with 'no'
+no
```

The regular expression /..? below would normally match the word 'no' and both ignore it as a term and exclude any phrases containing it. The above will allow it to appear as a term and in phrases.

```
# ignore all one or two-character words (unless =word appears below)
/..?
# ignore words with parenthesis, typically function() calls and the like
/.*\(.*
# ignore numbers, both cardinal (e.g. 1,234.0) and ordinal (e.g. 1st, 22nd)
/[0-9,.]+(st|nd|rd|th)?
```

These regular expressions ignore a lot of uninteresting terms that are typically code or other things that shouldn't be translated anyhow. There are many exceptions to the one or two-character word pattern in the default stoplist file, not only with = like '=in' but also '+no' and ':on' and '<ok' and '>of'.

```
# allow these words by themselves and don't count against length for phrases
:off
:on
```

These prepositions are common as button text and thus useful to have as terms; they also form an important part of phrases so are disregarded for term word count to allow for slightly longer phrases including them.

```
# allow these words by themselves, but ignore any phrases containing them
<first
<hello
<last</pre>
```

These are words that are worth including in a terminology, as they are common in applications, but which aren't generally part of idiomatic phrases.

```
# ignore these words by themselves, but allow phrases containing them
=able
=about
=actually
=ad
=as
=at
```

This is the largest category of stoplist words, and these are all just rather common words. The purpose of a terminology list is to provide specific translation suggestions for the harder words or phrases, not provide a general dictionary, so these words are not of interest by themselves, but may well be part of an interesting phrase.

```
# ignore these words by themselves, but allow phrases containing them, and
# don't count against length for phrases
#
# (possible additions to this list for multi-lingual text: >di >el >le)
#
>a
>an
>and
```

These very common words aren't of interest by themselves, but often form an important part of phrases so are disregarded for term word count to allow for slightly longer phrases including them.

```
# ignore these words and any phrases containing them
@ain't
@aint
@al
@are
```

These are "junk" words that are not only uninteresting by themselves, they generally do not contribute anything to the phrases containing them.

### pocount

pocount will count the number of strings and words in translatable files.

Supported formates include: PO and XLIFF. Almost all bilingual file formats supported by the Translate Toolkit will work with pocount, including: TMX, TBX, Gettext .mo, Qt .qm, Wordfast .txt TM.

A number of other *formats* should be countable as the toolkit develops. Note that only multilingual formats based the storage *base class* are supported, but that includes almost all storage formats.

### **Usage**

```
pocount [options] <directory|file(s)>
```

Where:

directory	will recurse and count all files in the specified directory
file(s)	will count all files specified

### Options:

**-h, --help** show this help message and exit

**--incomplete** skip 100% translated files

Output format:

**--full** (default) statistics in full, verbose format

--csv statistics in CSV format --short same as --short-strings

--short-strings statistics of strings in short format – one line per file
 --short-words statistics of words in short format – one line per file

# **Examples**

pocount makes it easy to count the current state of a body of translations. The most interesting options are those that adjust the output style and decide what to count.

# **Easy counting**

To count how much work is to be done in you project:

```
pocount project/
```

This will count all translatable files found in the directory project and output the results in --full format.

You might want to be more specific and only count certain files:

```
pocount *.po
```

This will count all PO files in the current directory but will ignore any other files that 'pocount' can count.

You can have full control of the files to count by using some of the abilities of the Unix commandline, these may work on Mac OS X but are unlikely to work on Windows.:

```
pocount $(find . -name "*.properties.po")
```

This will first find all files that match  $\star$ .properties.po and then count them. That would make it easy to count the state of your Mozilla translations of .properties files.

### Incomplete work

To count what still needs to be done, ignoring what is 100% complete you can use the --incomplete option.:

```
pocount --incomplete --short *.xlf
```

We are now counting all XLIFF files by using the  $\star$ .xlf expansion. We are only counting files that are not 100% complete and we're outputting string counts using the --short option.

### **Output formats**

The output options provide the following types of output

#### -full

This is the normal, or default, mode. It produces the most comprehensive and easy to read data, although the amount of data may overwhelm the user. It produces the following output:

```
avmedia/source/viewer.po
                strings
                             words (source)
                                               words (translation)
type
translated: 73465 ( 99%)
                                                   513296
                             538598 ( 99%)
               13 ( 0%)
                                141 ( 0%)
                                                      n/a
fuzzy:
                                602 ( 0%)
untranslated: 53 ( 0%)
                                                      n/a
             73531
                                                    513296
Total:
                             539341
```

A grand total and file count is provided if the number of files is greater than one.

#### -csv

This format is useful if you want to reuse the data in a spreadsheet. In CSV mode the following output is shown:

```
Filename, Translated Messages, Translated Source Words, Translated Target Words, 

→Fuzzy Messages, Fuzzy Source Words, Untranslated Messages, Untranslated Source 

→Words, Review Messages, Review Source Words 

avmedia/source/viewer.po, 1, 3, 3, 0, 0, 4, 22, 1, 3
```

Totals are not provided in CSV mode.

### -short-strings (alias -short)

The focus is on easily accessible data in a compact form. This will only count strings and uses a short syntax to make it easy for an experienced localiser to read.:

```
test-po/fuzzy.po strings: total: 1 | 0t 1f 0u | 0%t 100%f 0%u
```

The filename is followed by a word indicating the type of count, here we are counting strings. The total give the total string count. While the letters t, f and u represent 'translated', 'fuzzy' and 'untranslated' and here indicate the string counts for each of those categories. The counts are followed by a percentage representation of the same categories.

#### -short-words

The output is very similar to --short-strings above:

```
test-po/fuzzy.po source words: total: 3 | 0t 3f 0u | 0%t 100%f ∪ →0%u
```

But instead of counting string we are now counting words as indicated by the term 'source words'

### **Bugs**

- There are some miscounts related to word breaks.
- When using the short output formats the columns may not be exactly aligned. This is because the number of digits in different columns is unknown before all input files are processed. The chosen tradeoff here was instanteous output (after each processed file) instead of waiting for the last file to be processed.

### podebug

Insert pseudo translations or debug markers into target text in XLIFF, Gettex PO and other localization files.

The pseudo translation or debug markers make it easy to reference and locate strings when your translated application is running.

#### Use it to:

- Target your translations: see what files are being referenced for string appearing in your programs.
- · Debug translations: if you know in what file the message occurs then you can quickly find it and fix it.
- Check that everything is translatable: any English only text needs to be analysed so that it can be localised.
- Check for Unicode compliance: by inserting Unicode text outside of the Latin range it allows you to check that your program can handle non-Latin correctly.

### **Usage**

```
podebug [options] <in> <out>
```

#### Where:

<in></in>	is an input directory or localisation file	1
<out></out>	is an output directory or localisation file, if missing output will be to standard out.	1

#### Options:

**--version** show program's version number and exit

**-h, --help** show this help message and exit

**--manpage** output a manpage based on the help

- --progress=PROGRESS show progress as: dots, none, bar, names, verbose
- --errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback
- -i INPUT, --input=INPUT read from INPUT in po, pot formats
- -x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths
- -o OUTPUT, --output=OUTPUT write to OUTPUT in po, pot formats
- **-S, --timestamp** skip conversion if the output file has newer timestamp
- -f FORMAT, --format=FORMAT specify format string
- --rewrite=STYLE the translation rewrite style: xxx, en, blank, chef (v1.2), unicode (v1.2)
- --ignore=APPLICATION apply tagging ignore rules for the given application: kde, gtk, openoffice, libreoffice, mozilla

### **Formats**

A format string can have these various options:

f	full filename including directory
F	as %f but with .po file extension
b	base of filename
В	base of filename with .po file extension
d	directory name
S	preset OpenOffice.org modifier
c	use only consonants
h	hash value (since version 1.4 – see notes below)
N	a set number of characters

A format string may look like this:

- %cf the full filename without vowels
- [%10cb] the first ten character after compressing the base of the filename and place it in square brackets with a space before the real message
- [%5cd %cB] the first 5 consonants of the directory, followed by a dash then the consonants of the filename with a .po extension. All surrounded by square brackets with a space before the translations.
- %4h. insert a hash value of length 4

Complex format strings may make it too difficult to actually read the translation, so you are probably best served using as short a string as possible.

### Rewriting (style)

The rewriting options are designed to change the target text in various ways (c.f. the various *rewriting styles* available). This is mostly valuable for debugging English text. The 'xxx' rewriter is useful in that it allows you to identify text that has not localisable as that text will lack the xxx characters.

The 'en' rewriter can be used to prepare English hashed (see below) files for quickly finding strings that have spelling or other errors. It can also be used to create a translated English file which can then be used for other purposes such as British English translation.

### Ignoring messages

In some applications their are translations that should not be translated (usually these are configuration options). If you do translate them then the application will fail to compile or run.

The --ignore option allows you to specify the application for which you are producing PO debug files. In this case it will then not mark certain of the PO entries with debug messages.

In Mozilla we do not mark lone .accesskey, .width, .height, etc since these can really be thought of as configuration options.

### Hashing

Sometimes you find an error in a string. But it is difficult to search for the occurrence of the error. In order to make it easy to find a string in your files we can produce a hash on the strings location and other data. This produces unique alphanumeric sequences which are prepended to the target text. Thus now in your application you have your translated text and an alphanumeric value. Its is then easy to search for that value and find your problem string.

# **Usings podebug**

Here are some more examples in a series of blog posts.

#### -rewrite=STYLE

podebug allows you to rewrite the output text in a number of ways.

#### XXX

The target text is surrounded by xxx as follows

```
msgid "English"
msgstr "xxxEnglishxxx"
```

This is useful when you want to identify which text is localisable. There might be text in your application which you cannot localise this will allow you to quickly identify that text.

#### en

The source text is copied to the target

```
msgid "English"
msgstr "English"
```

In this way you can create translations that contain only the source text. Useful if you are preparing a roundtrip test or want to start an English derived translation such as British English. It produces the same results as msgen but with the advantage that you can add debug markers.

# blank

This simply empties your current translations

```
msgid "English"
msgstr ""
```

When you have a set of translation files but no template this allows you to essentially convert a PO into a POT file. This mimics the --empty functionality of msghack.

#### bracket

New in version 1.4.

Places brackets around the translated text.

```
msgid "English"
msgstr "[English]"
```

This can be used in the same way as  $x \times x$  to check for translatability. It is also useful with very long strings as it allows you to check that the full string in rendered and has not been cutoff by the application.

#### chef

New in version 1.2.

Rewrites the source text using mock Swedish as popularised by the Swedish Chef.

```
msgid "English"
msgstr "Ingleesh"
```

This is probably only useful for some fun. It's not guaranteed that every string will be rewritten as the mock Swedish rules might not apply thus its not ideal for identifying untranslatable strings.

# flipped

New in version 1.4.

Change the text into a version that uses equivalent Latin characters that are upside down.

```
msgid "English"
msgstr "uıs"
```

flipped can give an output that simulates RTL languages. It inserts RTL characters to try to achieve RTL-like results. Its not perfect but will give you some sense of whether your application can do RTL. Or just use it for fun!

For really testing right-to-left GUIs, you want to make sure that the whole application is shown in RTL, not just the strings. Test your pseudo-translated file as a translation of an RTL language like Arabic or Hebrew. In case the application relies on other files coming from libraries (like GTK+), you might need to repeat the process for them, or at least ensure that you have the Arabic/Hebrew .mo files for them installed.

# unicode

New in version 1.2.

Rewrites the source text with Unicode characters that looks like the Latin characters that they are replacing.

```
msgid "English"
msgstr "īş"
```

This allows a translator or programmer to test a programs ability to use Unicode message strings. By using characters in the Unicode range but that are related to the plain Latin characters that they replace we ensure that the messages are still readable.

**Note:** Before version 1.4, the rewrite rule will also rewrite variables and XML tags, which would cause problems in some situations. Run *pofilter* as a quick method to fix up incorrect changes, or upgrade to version 1.4.

### posegment

posegment takes a Gettext PO or XLIFF file and segments the entries, generating a new file with revised and smaller translation units.

This is useful for the creation of a file that can be used as a Translation Memory as you should get better matching after you have exposed translated sentences that might occur elsewhere in your work.

Posegment won't do very advanced sentence boundary detection and alignment, but has customisations for the punctuation rules of several languages (Amharic, Afrikaans, Arabic, Armenian, Chinese, Greek, Japanese, Khmer, Oriya, Persian). For the purpose of increasing your TM (as described below), it is already very useful. Give it a try and help us to improve it even more for your language.

### **Usage**

```
posegment [options] <input> <segmented>
```

#### Where:

<input/>	translations to be segmented
<segmented></segmented>	translations segmented at the sentence level

#### Options:

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

--progress=PROGRESS show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in po, pot, tmx, xlf formats

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in po, pot, tmx, xlf formats

-S, --timestamp skip conversion if the output file has newer timestamp

**-P, --pot** output PO Templates (.pot) rather than PO files (.po)

-l LANG, --language=LANG the target language code

**--source-language=LANG** the source language code (default 'en')

**--keepspaces** Disable automatic stripping of whitespace

**--only-aligned** Removes units where sentence number does not correspond

### **Examples**

You want to reuse all of your Pidgin translations in another Instant Messenger:

```
posegment pidgin-af.po pidgin-af-segmented.po
```

Now all of our Pidgin translation are available, segmented at a sentence level, to be used as a Translation Memory for our other translation work.

You can do the same at a project level. Here we want to segment all of our OpenOffice.org translation work, a few hundred files:

```
posegment af/ af-segmented/
```

We start with all our files in af which are now duplicated in af-segmented except files are now fully segmented.

#### Issues

- If the toolkit doesn't have segmentation rules for your language then it will default to English which might be incorrect.
- Segmentation does not guarantee reuse as your TM software needs to know how to segment when matching. If you use software that doesn't do segmentation, you can consider joining the original and the segmented files together with msgcat, to get the best of both worlds.
- You cannot (yet) use the tool to break a file into segments, translate, and then recreate as the segmented file does not know which parts should be joined together to recreate a file.

# pocompile

Compile PO or XLIFF files into MO (Machine Object) files. MO files are installed on your computer and allow a Gettext enabled computer to provide the translations for the application.

### Usage

```
pocompile <po> <mo>
```

#### Where:

<po xliff=""></po>	is a standard PO file, XLIFF file or directory
<mo></mo>	is the output MO file or directory of MO files

### Options:

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

--progress=PROGRESS show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in xlf, po, pot formats

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in mo format

**-S, --timestamp** skip conversion if the output file has newer timestamp

**--fuzzy** use translations marked fuzzy

**--nofuzzy** don't use translations marked fuzzy (default)

### **Examples**

```
pocompile --fuzzy file.po file.mo
```

Creates a new MO file called *file.mo* based on the translation in the PO file *file.po*. By using the --fuzzy option we use all translations including those marked fuzzy.

```
pocompile file.xlf file.mo
```

Create an MO file from an XLIFF file called *file.xlf* (available from version 1.1 of the toolkit).

#### poswap

This tool builds a new translation file with the target text (translation) of the input file(s) as source language of the output file it creates.

This makes it possible to have French as the source file for translation, rather than English. Note that this requires no change in the software project and is only a manipulation of the strings in the existing files. The only requirement for this tool is a French translation.

It can also be used to convert translatable files that use logical IDs instead of source text into a format usable by human localisers.

# **Usage**

```
poswap [options] <newsource> [-t current] <new>
```

#### Where:

<newsource></newsource>	is the translations (preferably 100% translated) of the preferred source language (like French)
<current></current>	is the (optional) current English based translation in your intended target language
<new></new>	is the intended output file / directory

### Options:

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

**--progress=PROGRESS** show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in pot format

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in po, pot formats

-t TEMPLATE, --template=TEMPLATE read from TEMPLATE in po, pot formats

**--reverse** Do the inverse operation (converting back to a normal English based file). See the examples.

# **Examples**

Ensure that the two po files / directories correspond 100% to the same pot file before using this.

To start a fresh Afrikaans (af) translation from Dutch (nl):

```
poswap nl.po nl-af.po
```

This initialises a new, empty file nl-af.po with Dutch as the source language.

To change the nl-af.po file back to the expected English based af.po:

```
poswap --reverse nl.po -t nl-af.po af.po
```

To translate Kurdish (ku) through French (fr):

```
poswap -i fr/ -t ku/ -o fr-ku/
```

This will take the existing (English based) Kurdish translation in ku/ and produce files in fr-ku with French as the source language and Kurdish as the target language.

To convert the fr-ku files back to en-ku:

```
poswap --reverse -i fr/ -t fr-ku/ -o en-ku/
```

This recreates the English based Kurdish translation from the French based files previously created in fr-ku/.

### **Issues**

- Behaviour is undetermined if the two files don't match 100%. If PO files are based in the same template, there should be no problem.
- We should probably be doing fuzzy matching in future to ease the migration over the lifetime of a changing French translation.

### poclean

This is a rudimentary tool to produce a clean file from an unclean file (Trados/Wordfast) by stripping out the tw4win indicators.

### **Usage**

```
poclean <input> <output>
```

Where:

<input/>	is the text versions of the unclean RTF files
<output></output>	is the intended output file / directory

# Options:

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

--progress=PROGRESS show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in pot format

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

- -o OUTPUT, --output=OUTPUT write to OUTPUT in po, pot formats
- -S, --timestamp skip conversion if the output file has newer timestamp

### **Examples**

To create a text version of the unclean RTF file, you need UnRTF, available here: project site or here (windows).

```
unrtf translation.rtf --text > translation.po
```

You might need to convert the encoding of the file, with icony, for example:

```
iconv -f latin1 -t utf-8 translation.po > new_translation.po
```

Now you can clean the file with poclean

```
poclean new_translation.po clean_translation.po
```

# pretranslate

Merge existing translations from an old translation file to a new one as well as fill any missing translations from translation memory via fuzzy matching.

This functionality used to be part of pot2po and corresponds to "msgmerge" from the gettext package. pretranslate works on PO and XLIFF files.

### **Usage**

```
pretranslate [options] <input> <output>
```

### Where:

<input/>	is the translation file or directory to be pretranslated
<output></output>	is the translation file or a directory where the pretranslated version will be stored

#### Options:

**--version** show program's version number and exit

-h, --help show this help message and exit--manpage output a manpage based on the help

--progress=PROGRESS show progress as: dots, none, bar, names, verbose

--errorlevel=ERRORLEVEL show errorlevel as: none, message, exception, traceback

-i INPUT, --input=INPUT read from INPUT in pot format

-x EXCLUDE, --exclude=EXCLUDE exclude names matching EXCLUDE from input paths

-o OUTPUT, --output=OUTPUT write to OUTPUT in po, pot formats

-t TEMPLATE, --template=TEMPLATE read old translations from TEMPLATE

**-S, --timestamp** skip conversion if the output file has newer timestamp

```
--tm=TM The file to use as translation memory when fuzzy matching
```

-s MIN\_SIMILARITY, --similarity=MIN\_SIMILARITY The minimum similarity for inclusion (default: 75%)

--nofuzzymatching Disable all fuzzy matching

### **Examples**

```
pretranslate -t zu-1.0.1 -tm zu_tm.po zu-2.0.2 zu-2.0.2-translated
```

Here we are pretranslating the PO or XLIFF files in zu-2.0.2 using the old translations in zu-1.0.1 and fuzzy matches from the zu-tm.po compendium. the result is stored in zu-2.0.2-translate

Unlike pot2po pretranslate will not change anything in the input file except merge translations, no reordering or changes to headers.

### Merging

It helps to understand when and how pretranslate will merge. The default is to follow msgmerge's behaviour but we add some extra features with fuzzy matching:

- · If everything matches we carry that across
- · We can resurrect obsolete messages for reuse
- If we cannot find a match we will first look through the current and obsolete messages and then through any global translation memory
- Fuzzy matching makes use of the *Levenshtein distance* algorithm to detect the best matches

### Performance

Fuzzy matches are usually of good quality. Installation of the python-Levenshtein package will speed up fuzzy matching. Without this a Python based matcher is used which is considerably slower.

### Levenshtein distance

The levenshtein distance is used for measuring the "distance" or similarity of two character strings. Other similarity algorithms can be supplied to the code that does the matching.

This code is used in *pot2po*, *tmserver* and Virtaal. It is implemented in the toolkit, but can optionally use the fast C implementation provided by python-Levenshtein if it is installed. It is strongly recommended to have **python-levenshtein** installed.

To exercise the code the classfile "Levenshtein.py" can be executed directly with:

```
$ python Levenshtein.py "The first string." "The second string"
```

**Note:** Remember to quote the two parameters.

The following things should be noted:

- Only the first MAX\_LEN characters are considered. Long strings differing at the end will therefore seem to match better than they should. A penalty is awarded if strings are shortened.
- The calculation can stop prematurely as soon as it realise that the supplied minimum required similarity cannot be reached. Strings with widely different lengths give the opportunity for this shortcut. This is by definition of the Levenshtein distance: the distance will be at least as much as the difference in string length. Similarities lower than your supplied minimum (or the default) should therefore not be considered authoritative.

# **Shortcommings**

The following shortcommings have been identified:

- Cases sensitivity: 'E' and 'e' are considered different characters and according differ as much as 'z' and 'e'. This is not ideal, as case differences should be considered less of a difference.
- **Diacritics:** 'ê' and 'e' are considered different characters and according differ as much as 'z' and 'e'. This is not ideal, as missing diacritics could be due to small input errors, or even input data that simply do not have the correct diacritics.
- Similar but different words: Words that have similar characters, but are different, could increase the similarity beyond what is wanted. The sentences "It is though." and "It is dough." differ markedly semantically, but score similarity of almost 85%. A possible solution is to do an additional calculation based on words, instead of characters.
- Whitespace: Differences in tabs, newlines, and space usage should perhaps be considered as a special case.
- tmserver a Translation Memory server, can be queried over HTTP using JSON
- poterminology extracts potential terminology from your translation files
- pocount Count words and strings in PO, XLIFF and other types of translatable files
- podebug Add debug strings to messages
- posegment Break a PO or XLIFF files into sentence segments, useful for creating a segmented translation memory
- pocompile create an MO (Machine Object) file from a PO or XLIFF file
- poswap uses a translation of another language that you would rather use than English as source language
- poclean produces a clean file from an unclean file (Trados/Wordfast) by stripping out the tw4win indicators
- pretranslate fill any missing translations from translation memory via fuzzy matching.
- Levenshtein distance edit distance algorithms for translation memory matching

# 1.5 Scripts

# 1.5.1 Mozilla L10n Scripts

#### Introduction

This page describes the purpose and usage of scripts available in the Translate Toolkit specifically for making the translation of Mozilla products easier.

Mozilla's move from CVS to Mercurial made a lot of these scripts necessary. For more information about Mozilla 110n from CVS, see the *moz-l10n-builder* page.

All of these scripts are available on Subversion from here.

We are currently generating POT files for most major betas, RCs and releases of Firefox and Thunderbird. They are available here: http://l10n.mozilla.org/pootle/pot/

As a start you might want to just use these POT files and gradually learn more about the processes described below. Contact us for more help on using these.

### Requirements

- The *Translate Toolkit* (>=1.3)
- All scripts in the tools/mozilla directory (from the project sources) should be executable and in your PATH.

# build\_ff3.1\_langs.sh

# **Description**

This is a simple bash script that embodies most of the Mozilla 110n process and does the following:

- 1. Update Mozilla sources
- 2. Update language files from Mozilla's L10n Mercurial repository.
- 3. Replace old 110n en-US files with a fresh copy from the updated source tree.
- 4. Create new POT files from the en-US 110n files.
- 5. Create archives of the POT files.
- 6. For each language:
  - 1. Update existing PO files if the checked out from a CVS, Subversion or Mercurial repository.
  - 2. Migrate PO files to new POT files.
  - 3. Create Mozilla 110n files for the language based on the migrated PO files.
  - 4. Create archives of the PO files.
  - 5. Build langpack for the language.

This script is used on the 110n.mozilla.org server to create most (if not all) of the files available from http://l10n. mozilla.org/pootle/. It was originally written as a stable way to provide these files and as such making it as general as possible was not the biggest requirement. This is evident in the script's very narrow focus.

#### Usage

This script takes no command-line parameters and is only configurable via the variables at the top and, failing that, custom hacking of the script.

The variables are used in the following ways:

1.5. Scripts 101

BUILD_DIF	The base build directory from where building is done.
MOZCENTRA	LThethirectory containing a checkout of the Mozilla source tree http://hg.mozilla.org/mozilla-central/
HG_LANGS	A space-separated list of language codes to build for.
L10N_DIR	The directory where Mozilla 110n files (from 110n-central) should be collected.
PO_DIR	The directory containing the externally-hosted or previously available source PO files (e.g. PO files
	managed in another VCS repository). It contains a sub-directory for each language.
POPACK_DI	RThe output directory for PO archives.
PORECOVER	
POT_INCLU	DASpace-separated list of files to be included in POT archives.
POTPACK_D	The output directory for POT archives.
POUPDATED	_The directory to use for updated PO files. It contains a sub-directory for each language.
LANGPACK_	DThe directory to put langpacks (XPIs) in.
FF_VERSIO	NThe version of Firefox that is being built for. This is used in the file names of archives.

**Note:** It is **strongly** recommended that you mirror the directory structure specified by the default values of the \*\_DIR variables. For example the default value for L10N\_DIR is \${BUILD\_DIR}/110n, then you should put your l10n-central check-outs in the 110n directory under your main build directory (BUILD\_DIR).

Basically, you should have an ideally separate build directory containing the following sub-directories: 110n, mozilla-central, po, popacks, potpacks, po-updated and xpi (if used). This way the only variable that need to be changed is BUILD\_DIR.

# build\_tb3\_langs.sh

This is the script that the build\_ff3.1\_langs.sh script above was actually adapted from. It is 90% similar with the obvious exception that it is aimed at building Thunderbird 3.0 packages in stead of Firefox 3.1. Also note that this script uses the comm-central repository in stead of mozilla-central.

#### buildxpi.py

### **Description**

Creates XPI language packs from Mozilla sources and translated 110n files. This script has only been tested with Firefox 3.1 beta sources.

It is basically the scripted version of the process described on Mozilla's "Creating a language pack" page.

This script is used by build\_ff3.1\_langs.sh to build language packs in its final step.

**Note:** This script uses the .mozconfig file in your home directory. Any existing .mozconfig is renamed to .mozconfig.bak during operation and copied back afterwards.

### **Usage**

```
buildxpi.py [<options>] <lang> [<lang2> ...]
```

Example:

```
buildxpi.py -L /path/to/110n -s /path/to/mozilla-central -o /path/to/xpi_output af ar
```

### Options:

```
-h, --help show this help message and exit
```

- -L L10NBASE, --l10n-base=L10NBASE The directory containing the <lang> subdirectory.
- **-o OUTPUTDIR, --output-dir=OUTPUTDIR** The directory to copy the built XPI to (default: current directory).
- **-p MOZPRODUCT, --mozproduct=MOZPRODUCT** The Mozilla product name (default: "browser").
- -s SRCDIR, --src=SRCDIR The directory containing the Mozilla 110n sources.
- **-d, --delete-dest** Delete output XPI if it already exists.
- **-v, --verbose** Be more noisy

#### get moz enUS.py

# **Description**

A simple script to collect the en-US 110n files from a Mozilla source tree ('comm-central' or 'mozilla-central') by traversing the product's 110n.ini file.

### **Usage**

```
get_moz_enUS.py [options]
```

### Options:

- **-h, --help** show this help message and exit
- -s SRCDIR, --src=SRCDIR The directory containing the Mozilla 110n sources.
- **-d DESTDIR, --dest=DESTDIR** The destination directory to copy the en-US locale files to.
- -p MOZPRODUCT, --mozproduct=MOZPRODUCT The Mozilla product name.
- **--delete-dest** Delete the destination directory (if it exists).
- **-v, --verbose** Be more noisy

#### moz-l10n-builder

This is the pre-Mercurial build script originally written by Dwayne Bailey. This is the script that all the others on this page replaces for post-CVS Mozilla 110n.

**Note:** This script is **not** applicable to the 110n process of any Mozilla products after the move to Mercurial.

For more information about this script see its dedicated page.

1.5. Scripts 103

# moz\_l10n\_builder.py

This script was intended to be a simple and direct port of the moz-l10n-builder script from above. It has pro's and cons in comparison to the original, but is very similar for the most part. So for more information about this script, see the original script's page.

# 1.5.2 moz-l10n-builder

Take a set of Mozilla (Firefox, Thunderbird, SeaMonkey, etc.) localisation and migrate them to the latest Mozilla source, building XPIs and repackaging hte Windows .exe file as needed.

Please also check the page on creating a language pack on the Mozilla wiki, to stay abreast of the latest Mozilla way of doing things.

**Note:** This page is only applicable to Mozilla products with its source hosted in CVS. This includes Firefox versions before 3.1 and Thunderbird versions before 3.0.

For information about working with the new source trees in Mercurial, see the *Mozilla L10n Scripts* page.

# **Prerequisites**

- Translation update component and building XPIs
  - Translate Toolkit
  - Existing Mozilla translations in PO format
  - A checkout of Mozilla sources updated to the correct BRANCH or RELEASE
- Building Windows executables
  - Firefox or Thunderbird en-US .exe file e.g. Firefox 2.0 en-US
  - upx for executable compression
  - Nullsoft installer to package the installer.
  - 7zip for various compression
  - Linux: WINE to run the Nullsoft installer
- Directory structure under the directory you want to run moz-110n-builder in:

110n/	Contains Mozilla 110n files for available/needed language(s)
mozilla/	The Mozilla source tree
po/	Contains your PO files (output from moz2po)
potpacks/	Where POT-archives go

Note these instructions are for building on Linux, they may work on Windows. All software should be available through your distribution. You will need to use Wine to install the Nullsoft installer and may need to sort out some path issues to get it to run correctly.

# **Latest Version**

moz-l10n-builer is not currently distributed as part of the toolkit. You can get the latest version from Git and you will also need this minor patch to the mozilla source code.

### **Usage**

```
moz-l10n-builder [language-code|ALL]
```

#### Where:

language-code	build only the supplied languages,	or build ALL if specified	or if no option is supplied

Your translations will not be modified.

### Operation

moz-110n-builder does the following:

- Updates the mozilla/ directory
- · Creates POT files
- Migrates your translations to this new POT file
- Converts the migrated POT files to .dtd and .properties files
- Builds XPI and .exe files
- Performs various hacks to cater for the anomalies of file formats
- Outputs a diff of you migrated PO files and your newly generated Mozilla 110n/ files

### **Bugs**

Currently it is too Translate.org.za specific and not easily configurable without editing. It is also not intelligent enough to work our that you want Firefox vs Thunderbird generation. A lot of this functionality should be in the Mozilla source code itself. We hope over time that this might happen.

# 1.5.3 phase

phase is a script that allows you to perform a number of tasks on a set of PO files that have been broken into phases. You can create a ZIP file for a phase, run checks against a phase, review a phase, edit files in a phase, etc. All the tasks that would be involved in sending work to various translators, receiving work, checking it and committing to CVS.

### **Prerequisites**

- An environment that will run bash
- diff
- cvs

### **Latest Version**

phase is not currently distributed as part of the toolkit. You can get the latest version from Git

1.5. Scripts 105

### Usage

```
phase <command> [options]
```

Mostly the usage follows the format of:

```
phase <command> <language-dir> <phaselist> <phase-name>
phase <command> <language-dir> <phase-name>
```

A full list of commands and options can be seen by running:

```
phase --help
```

### **Commands**

These are the commands that you can use:

- makephaselist <new-phase-list-name> creates a phase list
- listphases <phase-list> lists the different phases that appear in the phase-list file
- listfiles <phase-list> <phase-name> list all files for the given phase in the phase-list file
- checkphaselist <language-dir> <phase-list> checks to see which files are not included in the phaselist
- countpo <language-dir> <phase-list> <phase-name> counts PO file in the given phase
- countpot <template-dir> <phase-list> <phase-name> counts POT file in the given phase
- missingpo <language-dir> <phase-list> <phase-name> lists files that have not been returned for a phase
- packpot <template-dir> <phase-list> <phase-name> packs all POT files for a given phase into a ZIP file
- packpo <language-dir> <phase-list> <phase-name> packs all PO files for a given phase into a ZIP file
- packall <template-dir> <phase-list> packs all phases found in the phase list
- packallpo <language-dir> <phase-list> packs all phases found in the phase list for the given language
- countmismatch <language-dir> <template-dir> <phase-list> <phase-name> compares the source word count between PO and POT to determine if there are any file errors.
- editpo <language-dir> <phase-list> <phase-name> edit the PO files in a phase
- editpochecks <language> <phase-name> edit the PO checks output by checkpo
- editconflicts <language-dir> <phase-list> <phase-name> edit the extracted conflict items
- checkpo <language-dir> <phase-list> <phase-name> [pofilter options] run pofilter checks against the given phase
- mergepo <language> <phase-name> merge the checks back into the main language directory
- conflictpo <language-dir> <phase-list> <phase-name> [poconflict options] run poconflict checks against the given phase
- diffpo <language-dir> <phase-list> <phase-name> perform a cvs diff for the phase
- cvslog <language-dir> <phase-list> <phase-name> perform a cvs log against files in the phase
- lastlog <language-dir> <phase-list> <phase-name> retrieves the last cvs log entry for each file in a phase
- cvsadd <languages-dir> <phase-list> <phase-name> CVS adds files and directories that are not already in CVS

- diffpo <language-dir> <phase-list> <phase-name> perform a cvs diff for the phase
- reviewpo <language-dir> <phase-list> <phase-name> [pofilter options] extract items marked for review for the given phase
- editreviews <language-dir> <phase-list> <phase-name> edit the extracted review items
- countreviews < language-dir> < phase-list> < phase-name> count the number of strings and words under review
- checkinpo <language-dir> <phase-list> <phase-name> cvs checkin the files in the given phase
- creategsi <language-dir> <en-US.gsi> <traget-language> creates a BZ2 GSI/SDF file for the language against the en-US GSI file
- reviewsinout <language> <phase-name> counts the number of review files returned vs sent and shows which are missing
- reviewsdiff <language> <phase-name> create a diff between what was sent for review and what was returned

### **Bugs**

There are probably lots mostly the bug is that the command line options are pretty inconsistent

# 1.5.4 pocompendium

Takes a directory of translated PO files and creates a single PO files called a PO compendium. This compendium can be used to review word choice conflicts or as input during a merge using *pomigrate2*.

## **Prerequisites**

### **GNU Gettext:**

- msgattrib
- · msgcat
- msghack (may not be present on your installation of Gettext, but is only required for the invert command)
- · msgfilter

#### **Usage**

```
pocompendium [options] output.po <-d po-directory(ies)|po-file(s)>
```

### Where:

output.po the name of the output PO compendium	
po-directory(ies)	one or more directories to use as input for the compendium
po-file(s)	one or more PO files to use as input for the compendium

#### Options:

-v, --invert swap the msgid and msgstr in the input PO files
 -e, --errors only return those msg blocks that have conflicts
 -i, --ignore-case drops all msgstr's to lowercase

1.5. Scripts 107

```
    -st, -tilde, --strip-accel-amp remove all & style accelerator markers
    -sa, -amp, --strip-accel-tilde remove all ~ style accelerator markers
    -su, --strip-accel-under remove all _ style accelerator markers
```

### **Examples**

- Compendium creation create a compendium with all your translations to use as input during a message merge either when migrating an existing project or starting a new one.
- Conflicting translations use --errors to find where you have translated an English string differently. Many times this is OK but often it will pick up subtle spelling mistakes or help you to migrate older translations to a newer choice of words
- Conflicting word choice use --invert and --errors to get a compendium file that show how you have used a translated word for different English words. You might have chosen a word that is valid for both of the English expressions but that in the context of computers would cause confusion for the user. You can now easily identify these words and make changes in the underlying translations.

### **Narrowing Results**

PO files treat slight changes in capitalisation, accelerator, punctuation and whitespace as different translations. In cases 2) and 3) above it is sometimes useful to remove the inconsistencies so that you can focus on the errors in translation not on shifts in capitals. To this end you can use the following:

```
--ignore-case, --strip-accel-amp, --strip-accel-tilde, --strip-accel-under
```

### Operation

pocompendium makes use of the Gettext tool msgcat to perform its task. It traverses the PO directories and cat's all found PO files into the single compendium output file. It then uses msgattrib to extract only certain messages, msghack to invert messages and msgfilter to convert messages to lowercase.

### **Bugs**

There are some absolute/relative path name issues

# 1.5.5 pocommentclean

pocommentclean will remove all translator comments from a directory of PO files.

### **Prerequisites**

sed

### **Usage**

```
pocommentclean [--backup] <po>
```

Where:

no	is a directory of existing PO files that you want to clean
PO	is a directory of existing 10 lifes that you want to clean

Options:

**--backup** Create a backup file for each PO file converted, .po.bak

## Operation

Using sed pocommentclean will delete all lines starting with # but which are not standard Gettext PO format lines. So it won't delete developer comments (#.), obsolete messages (#~), flags (#,) or locations (#:).

### **Bugs**

pocommentclean cannot clean individual PO files, it only cleans directories

# 1.5.6 pomigrate2

pomigrate2 aims to move an existing translation to a new version based on updated PO Template files automatically without user intervention. Therefore it is ideal for when you are migrating many languages or migrating from related but divergent products e.g. Mozilla to Firefox.

## **Prerequisites**

**GNU Gettext:** 

- msginit
- · msgcat
- msgmerge

### **Usage**

```
pomigrate [options] <from> <to> <new templates>
```

### Where:

from	is a directory of existing PO files	
to	is the directory where the migrated PO files will be stored	
new templates	this is the directory that contains the PO Template files	

## Options:

- -F, --use-fuzzy-matching use fuzzy algorithms when merging to attempt to match strings
- -C, --use-compendium create and use a compendium built from the migrating files
- -C, --use-compendium=COMPENDIUM use an external compendium during the migration
- **--no-wrap** do not wrap long lines
- **--locale** set locale for newly born files

1.5. Scripts 109

**-q, --quiet** suppress most output

-p, --pot2po use pot2po instead of msgmerge to migrate

## Operation

pomigrate2 makes use of the Gettext tools msgmerge or Translate Toolkit's pot2po to perform its merging tasks.

It firstly finds all files with the same name and location in the <from> directory as in the <template> directory and copies these to the <to> directory. If there is no file in the <from> directory to match one needed by the <template> directory then it will msgcat all files in the <from> directory with the same name and copy them to the correct destination in the <to> directory. If all of that fails then msginit is used to initialise any missing PO files.

Lastly all the files in <to> are merged using msgmerge or pot2po. This process updates the files to match the layout and messages in <templates>. Optionally, by using --use-compendium, a compendium of all the translations in <from> can be created to be used in the final merge process.

# 1.5.7 popuretext

Extracts all the source text from a directory of POT files or the target text from a directory of PO files, removing PO headers and optionally the accelerator keys.

If you want to use other tools to analyse the text within a translation project, then this is the tool for you. For example, you can use it to calculate word frequencies to create an initial glossary based on the pure source text.

## **Prerequisites**

- GNU Gettext
- sed

## **Usage**

```
popuretext <-P pot-dir|po-dir> <file.txt> [accelerator]
```

### Where:

pot-dir	a directory containing POT files
po-dir	a directory containing PO files
file.txt	file that contains the output text
accelerator	optional: accelerator marker to be removed from the text

# **Examples**

```
popuretext -P pot pot.txt '&'
```

Extract all the source text from the pot directory and place it in the pot.txt file removing all occurrences of the & accelerator.

```
popuretext af af.txt
```

Extract all target text from the Afrikaans files in the af directory, placing the extracted text in af.txt. In this case we are not filtering any accelerator characters.

# 1.5.8 poreencode

Takes a directory of existing PO files and converts them to a given encoding.

# **Prerequisites**

**GNU Gettext** 

### **Usage**

```
poreencode <encoding> <PO directory>
```

Where:

encoding	is the encoding you would like to convert to e.g. UTF-8
PO directory	is a directory of existing PO files

It is best to backup files before the conversion or to perform it against CVS which prevents a potential loss of data.

# **Operation**

poreencode makes use of the Gettext tool msgconv to perform its task. It traverses the PO directory and finds all PO file. It uses msgconv to convert the PO file from its existing encoding to the new encoding.

### **Bugs**

Like most Gettext tools they do a little bit more than documented, msgconv will decide which strings are in fact fuzzy and delete fuzzy marking – not a lot but you do need to diff (this probably related to #, fuzzy entries that are not placed in the place Gettext expects them).

# 1.5.9 posplit

Takes an existing PO file and splits it into three components: translated, untranslated and fuzzy. This is useful for reviewing translations or for extracting good translations from a compendium file.

Note that the input file is removed by the script (until version 1.9.1). The generated output files can be combined again with msgcat.

# **Prerequisites**

**GNU Gettext** 

1.5. Scripts 111

### **Usage**

```
posplit ./file.po
```

Where:

file.po	is an existing PO file or PO compendium

## **Bugs**

- Some relative path bugs thus the need for ./ before file.po.
- Until version 1.9.1, the original input file was removed, issue 2006.

The scripts are for working with and manipulating PO files. Unlike the tools which are written in Python, the scripts are written in bash. Some of them are packaged since version 1.0 of the Toolkit, but you might need to download them from version control and do a manual installation.

- moz-l10n-builder Create Mozilla XPIs and rebuild Windows installers from existing translations
- Mozilla L10n Scripts Build Mozilla products Firefox and Thunderbird
- phase Helps manage a project divided into phases of work, including sending, checking, etc
- pocompendium Creates various types of PO compendium (i.e. combines many PO files into a single PO file)
- pocommentclean Remove all translator comments from a PO file
- pomigrate2 Migrate older PO files to new POT files
- popuretext Extracts all the source text from a directory of POT files
- poreencode Converts PO files to a new character encoding
- posplit Split a PO file into translate, untranslated and fuzzy files

## 1.6 Use Cases

# 1.6.1 Migrating your translations

You very often need to migrate older translations to newer template or POT files. There are a number of Gettext tools that can manage this but they do not handle the situation where files have been renamed and moved. The *pomigrate2* script allows us to migrate between versions where there has been considerable change.

This migration HOWTO takes you through the steps in a generic fashion so that you can apply it to any of your projects. We use OpenOffice.org as an example for clarity. Our task in the examples is to migrate old translation for OpenOffice.org 1.1.3 to OpenOffice.org 2.0.

### Requirements

You will need:

- pomigrate2
- pocompendium
- · A text editor

· A PO editing tool

## Preparing the new POT files

We need the new POT files. Either download these from the project or generate them using *moz2po*, *oo2po* or the other tools of the Translate Toolkit. The POT files are templates for the destination files that we will be creating.

```
oo2po -P en-US.sdf ooo-20-pot
```

This will create new POT files in ooo-20-pot.

### Checking your old PO files for errors

We will be migrating your old PO files into the new POT files. This is a good opportunity to check for encoding errors and inconsistencies.

We use *pocompendium* to check for encoding errors:

```
pocompendium check.po -d ooo-113-old
```

This will create a compendium PO files, *check.po*, from all the PO files in the directory *ooo-113-old*, where *ooo-113-old* contains all your old translations. pocompendium is a wrapper around various Gettext tools, encoding errors will appear as errors from those tools.

Use your text editor to find and correct these errors. If you do not correct these now they will migrate to your new version. Once encoding errors are fixed they're usually gone for good, so it is time well spent.

### Optional: Checking your old PO files for consistency

**Note:** Note this step is optional, a more detailed explanation is given in *Checking for inconsistencies in your translations*.

We now look at consistency within the translations. The first check extracts situations were the same English string was translated in two different ways:

```
pocompendium --ignore-case --accel-amp --errors check.po -d ooo-113-old
```

In *check.po* you will find all situations where the same English text was translated differently. We use <code>--accel-amp</code> to remove accelerator markers (you'll change this depending on the one used by the project – we can do & \_ or ~). Now view *check.po* in a PO editor or text editor. You will need to correct each inconsistency in the source PO files, using *check.po* as the guide. Many of the errors are usually spelling mistakes. You can regenerate *check.po* from time to time until all inconsistencies are justified or removed.

Then we check for words in your language that are used for more than one English concept. You don't for instance want the same word for *Cancel* and *Delete*. For this we invert the compendium:

```
pocompendium --invert --ignore-case --accel-amp --errors check.po -d ooo-113-old
```

We now have a file similar to the previous one except your language appears in the msgid and the English appears in the msgstr. Look for inconsistencies that would cause problems for the user and correct them in the source files.

### **Migrate**

You are now ready to migrate using *pomigrate2*. You have created your destination POT files and all your PO files are clean and ready to migrate.

```
pomigrate2 ooo-113-old ooo-20-new ooo-20-pot
```

This will take all translations from *ooo-113-old* and migrate them to *ooo-20-new* using *ooo-20-pot* as templates. By default pomigrate2 migrates without any fancy text matching, there are options to allow for fuzzy matching and the use of a compendium. Read the *pomigrate2* help page to find out more about these options.

## Techie: what does pomigrate2 do to your file?

This section is for those insanely curious about what pomigrate will do to their files. You don't need to understand this section:-)

- Init stage
  - If a file has not changed location between old and new then it is simply copied across
  - If it has moved then we try to find a file by the same name and move ours there. If there are multiple files
    by the same name, then we join them together and copy them
  - If a file does not exist then we initialise it
- Update stage
  - We now update our translations using msgmerge or pot2po
  - If you asked for a compendium, we will build one from the existing files and update using it and optionally other external compendiums

That's it. At the end you should have every file that needs translation updated to the latest template files. Files that moved should still be preserved and not lost. Files that where renamed will still be translated if you used a compendium otherwise they will be untranslated.

## How well did you do

Congratulations! Your files are now migrated.

You might want to see how much of your old work was reusable in the new version:

```
pocount ooo-20-new
```

This will use *pocount* to count the words in your new files and you can compare the number of translate and untranslated messages from your old version.

### Conclusion

Your files have now been migrated and are ready for updating. If files have been moved or renamed, and you used a compendium, then most likely you have most of that work translated.

# 1.6.2 Checking your files with PO filter

*pofilter* allows you to check your PO or XLIFF files for certain common errors. This quick-start guide takes you through the process of using this tool, making corrections and merging your correction back into your translations.

The toolkit also other tools that can assist with quality assurance.

#### Quickstart

*Use any preferred text editor wherever* vim *is used.* 

- 1. Select filter(s): pofilter -1
- 2. Run filter(s): pofilter -i existing\_files/ -o errors/ [-t specific tests] [--excludefilter don't perform specific tests]
- 3. Delete items you don't want changed, set fuzzy if needed, delete if not needed: vim errors/\*.po
- 4. Merge changes back: pomerge -i errors/ -o existing\_files/ -t existing\_files/ (will overwrite existing files)
- 5. Create a patch for the changes: cvs diff -u existing\_files/ > x.diff
- 6. Check to see that the updates are what you want: vim x.diff
- 7. Commit changes: cvs ci existing\_files/

## **Detailed Description**

*pofilter* runs a number of checks against your translation files. Any messages that fail are output to a set of new files (in the same structure as the source/input files). You then edit these new/output files to correct any errors. Once you are satisfied with your corrections these corrected files are then merged back into the original files using *pomerge*.

# **Extracting Errors**

pofilter will run all tests unless you use the -t or --excludefilter options. There are over 38 tests and pofilter can itself provide you with a current list of all the available checks:

```
pofilter -l
```

We want to run the: accelerators, escapes, variables and xmltags tests as these are the ones most likely to break programs at runtime. We are also working with OpenOffice.org PO files created using *oo2po* so we want to ensure that we set the accelerator key marker and variables definitions correctly:

```
pofilter -t accelerators -t escapes -t variables -t xmltags --openoffice existing_ \hookrightarrow files errors
```

Any messages that fail one of the 4 checks will be placed in files in *errors*. We also used the —openoffice option to ensure that the tool is aware of the OpenOffice.org accelerator marker (~) and the OpenOffice.org variable styles (OpenOffice.org has over 10 variable styles). You can also specify other styles of project including GNOME, KDE or Mozilla.

You can also specify whether you want fuzzy entries included and checked, by specifying the --fuzzy parameter. By default this is off because fuzzy strings are usually known to be broken and will be reviewed by translators anyway.

Similarly you can include items marked for review by specifying --review or --ingnorereview. By default review items are included. This is not part of the standard Gettext format. We have allowed entries like this when we want to communicate to someone what error we have picked up:

```
# (review) - wrong word for gallery chosen
```

You can run pofilter without the -t option. This runs all the checks. This can be confusing if you have a lot of errors as you easily lose focus. One strategy is to run each test individually. This allows you to focus on one problem at a time across a number of files. It is much easier to correct end punctuation on its own then to correct many different types of errors. For a small file it is probably best to run all of the test together.

By using the --autocorrect option you can automatically correct some very common errors. Use with caution though. This option assumes you use the same punctuation style as the source text.

### **Edit the files**

Once the errors have been marked you can edit them with any text editor or PO editor e.g. Virtaal. You will be editing the files in the *errors* directory. Only messages that failed one of the tests will be present. If no messages failed then there will be no error PO file for the source PO file. Only critical errors are marked fuzzy – all others are simply marked with the pofilter marker. Critical errors are marked fuzzy as this allows you to simply merge them back into you PO files and then rely on the fact that all po2\* tools will ignore a message marked fuzzy. This allows you to quickly eliminate messages that can break builds.

To edit run:

```
vi `find errors -name "*.po"`
virtaal `find errors -name "*.po"`
```

or similar command.

The pofilter marker helps you determine what error was discovered:

```
# (pofilter) <test> - <explanation of test error>
```

Use the test description to help you determine what is wrong with the message. Remember that all your changes will be ported back into the PO files. So if you leave a string fuzzy in the error files, it will become fuzzy in the main files when you merge the corrected file back into the main file. Therefore delete anything you do not want to migrate back when you merge the files. Delete the test comments and fuzzy markings as needed. Leave them in if you want another translator to see them.

The computer can get it wrong, so an error that pofilter finds may in fact not be an error. We'd like to hear about these false positives so that we can improve the checks. Also if you have some checks that you have added or ideas for better checks, then let us know.

## Merging your corrections back into the originals

After correcting the errors in the PO files its time to merge these corrections back into the originals using *pomerge*.

```
pomerge -t existing_files -i errors -o files_without_errors
```

If -t and -o are the same directory, the corrections will be merged into the existing files. Do this only if you are using some kind of version control system so that you can check the changes made by *pomerge*.

## Checking the corrections

We have done this against CVS but you could run a normal diff between a good copy and your modifications. Thus we assume in the last step that we merged the corrections into the existing translations:

```
pomerge -t existing_files -i errors -o existing_files
```

Now we check the changes using cvs diff:

```
cvs diff -u existing_files > x.diff
```

This creates a unified diff (one with + and - lines so you can see what was added and what was removed) in the file x.diff:

```
vim x.diff
```

Check the diff file in any editor, here we use vim. You should check to see that the changes you requested are going in and that something major did not go wrong. Also look to see if you haven't left any lines with "# (pofilter): test description" which should have been deleted from the error checking PO files. Also check for stray fuzzy markers that shouldn't have been added. You will have to make corrections in the files in *existing\_files* not in *errors*.

When you are happy that the changes are correct run:

```
cvs ci existing_files
```

Congratulations you have helped eliminate a number of errors that could give problems when running the application. Now you might want to look at running some of the other tests that check for style and uniformity in translation.

# 1.6.3 Using csv2po

csv2po allows you to create CSV files from PO files. This allows you to send translation work to translators who do not or cannot use PO Editors but who can use a Spreadsheet.

### Quickstart

- 1. pofilter --fuzzy --review -t untranslated <po-dir> <po-filtered-dir> (this step is optional)
- 2. divide into sections
- 3. po2csv <po-dir|po-filtered-dir> <csv-out>
- 4. edit in Excel or OpenOffice.org Calc
- 5. csv2po --charset=windows-1250 -t templates <csv-in> <po-in> (you must work against a template directory, the charset option corrects problems with characters sets)
- 6. /commands/phase to do basic checks sort out encoding issues
- 7. pomerge --mergeblank=no -t <po-dir> <po-in> <po-dir>
- 8. git diff check the changes
- 9. git add & git commit commit changes

## **Detailed Description**

po2csv allows you to send CSV files, which can be edited in any spreadsheet, to a translator. This document outlines the process to follow from the raw po files -> CSV files -> back to PO. We also look at a case where you may have submitted a subset of the PO files for translation and you need to integrate these.

## Creating a subset

This step is optional.

To send a translator only those messages that are untranslated, fuzzy or need review run:

```
pofilter --isfuzzy --isreview -t untranslated <po-dir> <po-filtered-dir>
```

### **Divide into sections**

You might want to divide the work into sections if you are apportioning it to different translators. In that case create new directories:

```
e.g. po-filtered-dir-1 po-filtered-dir-2

or po-filtered-dir-bob po-filtered-dir-mary
```

Copy files from *po-filtered-dir* to *po-filtered-dir-N* in a way that balance the work or apportions the amounts you want for each translator. Try to keep sections together and not break them up to much e.g. Give one translator all the OpenOffice.org Calc work don't split it between two people – this is just a simple measure to ensure constancy.

Now continue as normal and convert to CSV and perform word counts for each separate directory.

### Creating the CSV files

```
po2csv <po-dir|po-filtered-dir> <csv-out>
```

This will create a set of CSV files in *csv-out* which you can compress using zip.

# Creating a word count

Professional translators work on source word counts. So we create a word count to go with the file:

```
pocount `find po-dir|po-filtered-dir -name "*.po"`
```

We work on source words regardless of whether the string is fuzzy or not. You might want to get a lower rate for work on fuzzy strings.

Place the word count file in both the PO and CSV directory to avoid the problem of finding it later. Check the number to make sure you haven't inadvertently including something that you didn't want in.

### Package the CSV files

```
zip -r9 work.zip <csv-out>
```

## **Translating**

Translators can use most Spreadsheets. Excel works well. However there are a few problems with spreadsheets:

- Encoding you can sort that out later
- Strings that start with '- most spreadsheets treat cells starting with 'as text and gobble up the '. A work around is to escape those like this '. po2csv should do this for you.
- Autocorrect Excel changes ... to a single character and does other odd things. pofilter will help catch these
  later.
- Sentences with + or +- will create errors and the translators will have to escape them as + +-
- Sentences that only contain numbers can get broken: "1." will be converted to "1"

# Converting Excel spreadsheets to CSV file

You can, and should, keep your files as CSV files. However, many translators are not the best wizzes at using their spreadsheet. In this case many files will have been changed to XLS files. To convert them by hand is tedious and error prone. Rather make use of xlHtml which can do all the work for you.

```
xlhtml -xp:0 -csv file.xls > file.csv
```

## Converting CSV back to PO

Extract the CSV files here we assume they are in *csv-in*:

```
csv2po --charset=windows-1250 -t <templates> <csv-in> <po-in>
```

This will create new PO files in *po-in* based on the CSV files in the *csv-in* and the template PO files in *templates*. You shouldn't run the csv2po command without templates as this allows you to preserve the original file layout. Only run it without -t if you are dealing with a partial part of the PO that you will merge back using a *pomerge*.

**Note:** Running csv2po using the input PO files as templates give spurious results. It should probably be made to work but doesn't

Note: You might have encoding problems with the returned files. Use the --charset option to convert the file from another encoding (all PO files are created using UTF-8). Usually Windows user will be using something like WINDOWS-1250. Check the file after conversion to see that characters are in fact correct if not try another encoding.

## Checking the new PO files

Use *pofilter* to run checks against your new files. Read *Checking your files with PO filter* to get a good idea of how to use the tool.

## **Removing fuzzies**

When you merge work back that you know is good you want to make sure that it overrides the fuzzy status of the existing translations, in order to do that you need to remove the "#, fuzzy" markers.

This is best performed against CVS otherwise who knows what changed.

### Merging PO files into the main PO files

This step would not be necessary if the CSV contained the complete PO file. It is only needed when the translator has been editing a subset of the whole PO file.

```
pomerge --mergeblank=no -t po-dir -i po-in -o po-dir
```

This will take PO files from *po-in* merge them with those in *po-dir* using *po-dir* as the template – i.e. overwriting files in *po-dir*. It will also ignore entries that have blank msgstr's i.e. it will not merge untranslated items. The default behaviour of pomerge is to take all changes from *po-in* and apply them to *po-dir* by overriding this we can ignore all untranslated items.

There is no option to override the status of the destination PO files with that of the input PO. Therefore all your entries that were fuzzy in the destination will still be fuzzy even thought the input was corrected. If you are confident that all your input is correct then relook at the previous section on removing fuzzies.

# 1.6.4 Creating OpenOffice.org POT files

This quick start guide shows you how to create the PO Template files for your OpenOffice.org translation.

#### **Quick Start**

- 1. Download the latest POT and GSI files
- 2. oo2po -P <gsi> <new-pots>

### **Detailed Description**

#### Download the latest POT and GSI files

The POT files produced by Pavel Janik contain the associated en-US.sdf file that you need to create your own languages SDF file. This is the same file that produces the POT files. So to begin translating you don't need to go further than this.

• Download the latest POT and GSI files

However, you will need this file if you need to use some of the other features of *oo2po* such as changing the source language from English.

### Produce the POT files using oo2po

```
oo2po -P <gsi> <new-pots> oo2po -P en-US.gsi pot
```

This takes the *en-US.gsi* file and creates POT files in the *pot* directory. The  $\neg P$  option ensures that .pot files are created instead of .po file.

If you want to create one large .pot file instead of a lot of small ones, you should use the:

```
oo2po -P --multifile=onefile en-US.gsi pot
```

option as described in oo2po.

### Produce a POT files with French source text

You will need to have access to a French GSI file. The following commands will create a set of POT files with French as the source language:

```
oo2po -P --source-language=fr fr.gsi pot-fr
```

This will take translations from fr.gsi and create a set of POT files in pot-fr. These POT files will have French as the source language. You need to make sure that fr.gsi is in fact up to date.

# 1.6.5 Checking for inconsistencies in your translations

Over time language changes, hopefully not very quickly. However, if your language is new to computers the change might be rapid. So now your older translations have different text to your new translations. In this use case we look at how you can bring alignment back to your translations.

Other cases in which you can expect inconsistencies:

- Multiple translators are involved
- Translations are very old
- You prepared this set of translations with translations from multiple sources
- You changed terminology at some stage in the translation
- You did not do a formal glossary development stage

### What we won't be able to achieve

We cannot find grammatical errors and we won't be able to find all cases of words, etc

### Scenario

You are translating Mozilla Firefox into Afrikaans. The files are stored in af. You have the following issues:

• Your current translator is good but took over from a team of three

• Terminology is well defined but not well used by the old translators

We'll look at the translations first from the English, or source text, point of view. Then we will look at it from the Afrikaans point of view. The first will pick up where we have translated the same English word differently in Afrikaans i.e. an inconsistency. While the second will determine if we use the same English word for different English words, possibly this will confuse a user.

### Step 1: Extracting conflicting target text translations

```
poconflicts -I --accelerator="&" af af-conflicts
```

From our existing translation in af we extract conflicts and place them in af-conflicts. We are ignoring case with -I so that Save as is considered the same as Save As. The --accelerator options allows us to ignore accelerators so that File is the sane as &File which is also the same as Fi&le

If we browse into *af-conflicts* we will see a flat structure of words with conflicts.

\$ cd af-conflicts \$ 1s					
change.po	disc.po	functionality.po	letter.po	overwrite.po	u
→ restored.po changes.po	document.po	gb.po	library.po	page.po	u
→ restore.po character.po	dots.po	graphic.po	light.po	pager.po	u
retry.po chart.po	double.po	grayscale.po	limit.po	percent.po	
→ return.po check.po	down.po	grid.po	line.po	pies.po	
→ right.po	drawing.po	group.po	-		
etc		5 <u>-</u>			

These are normal PO files which you can edit in any PO editor or text editor. If we look at the first file change.po we can see that the source text *Change* was translated as *Verander* and *Wysig*. The translators job is now to correct these PO files, ignoring instances where the difference is in fact correct.

Once all fixes have been made we can merge our changes back into the original files.

# Step 2: Merging our corrections back into the original files

Our files in af-conflicts are in a flat structure. We need to structure them into the hierarchy of the existing PO files.

```
porestructure af-conflicts af-restructured
```

The entries that where in the files in *af-conflicts* have been placed in *af-restructured*, they now appear in the correct place in the directory structure and also appear in the correct file. We are now ready to merge.

```
pomerge -t af -i af-restructure -o af
```

Using the existing files in *af* we merge the corrected and restructured file from *af-restructure* and place them back into *af*. Note: use a different output directory if you do not want to overwrite your existing files. All your conflict corrections are now in the correct PO file in *af*.

You might want to run Step 1 again to make sure you didn't miss anything or introduce yet another problem.

Next we look at the inverted conflict problem.

## Step 3: Extracting conflicts of meaning

If you have used the same Afrikaans word for two different English words then you could have created a conflict of meaning. For instance in our Xhosa translations the word Cima was used for both Delete and Cancel. Clearly this is a serious issue. This step will allow us to find those errors and take action.

```
poconflicts -v -I --accelerator="&" af af-conflicts-invert
```

We use the same command line as in **Step 1** but add  $\neg v$  to allow us to invert the match. We are also now outputting to *af-conflicts-invert* to make things clear.

This time the PO files that are created have Afrikaans names

```
$ cd af-conflicts-invert
$ 1s
dataveld.po
                                             lisensieooreenkoms.po paragraaf.po
                         grys.po

→ sny.po

datumgekoop.po
                         hallo.po
                                             lysinhoud.po
                                                                    pasmaak.po
→ soek.po
                         hiperboliese.po
datum.po
                                             maateenheid.po
                                                                    persentasie.po
→ sorteer.po
deaktiveer.po
                         hoekbeheer.po
                                             maatskappynaam.po
                                                                    posadres.po
→ sorteervolgorde.po
etc...
```

We edit these as usual. You need to remember that you will see a normal PO file but that you are looking at how the translation might be confusing to a user. If you see the same Afrikaans translation for two different English terms but there is no conflict of meaning or no alternative then leave it as is. You will find a lot of these instances so the results are less dramatic then the results from a normal conflict analysis.

Lastly follow Step 2 to restructure and merge these conflicts back into your translations

### Conclusion

You've now gone a long way to improving the quality of your translations. Congratulations! You might want to take some of what you've learnt here to start building a terminology list that can help prevent some of the issues you have seen.

# 1.6.6 Creating a terminology list from your existing translations

If you did not create a terminology list when you started your translation project or if you have inherited some old translations you probably now want to create a terminology list.

A terminology list or glossary is a list of words and phrases with their expected translation. They are useful for ensuring that your translations are consistent across your project.

With existing translations you have embedded a list of valid translation. This example will help you to extract the terms. It is only the first step you will need to review the terms and must not regard this as a complete list. And of course you would want to take your corrections and feed them back into the original translations.

### **Quick Overview**

This describes a multi-stage process for extracting terminology from translation files. It is provided for historical interest and completeness, but you will probably find that using *poterminology* is easier and will give better results than following this process.

- Filter our phrases of more than N words
- Remove obviously erroneous phrases such as numbers and punctuation
- Create a single PO compendium
- Extract and review items that are fuzzy and drop untranslated items
- Create a new PO files and process into CSV and TMX format

### Get short phrases from the current translations

We will not be able to identify terminology within bodies of text, we are only going to extract short bit of text i.e. ones that are between 1 and 3 words long.

```
pogrep --header --search=msgid -e '^\w+(\s+\w+){0,2}$' zulu zulu-short
```

We use --header to ensure that the PO files have a header entry (which is important for encoding). We are searching only in the msgid and the regular expression we use is looking for a string with between 1 and 3 words in it. We are searching through the folder *zulu* and outputting the result in *zulu-short* 

### Remove any translations with issues

You can for instance remove all entries with only a single letter. Useful for eliminating all those spurious accelerator keys.

```
pogrep --header --search=msgid -v -e "^.$" zulu-short zulu-short-clean
```

We use the -v option to invert the search. Our *cleaner* potential glossary words are now in *zulu-short-clean*. What you can eliminate is only limited by your ability to build regular expressions but yu could eliminate:

- Entries with only numbers
- Entries that only contain punctuation

#### Create a compendium

Now that we have our words we want to create a single files of all terminology. Thus we create a PO compendium:

```
~/path/to/pocompendium -i -su zulu-gnome-glossary.po -d zulu-short-clean
```

You can use various methods but our bash script is quite good. Here we ignore case, -i, and ignore the underscore (\_) accelerator key, -su, outputting the results in.

We now have a single file containing all glossary terms and the clean up and review can begin.

# Split the file

We want to split the file into translated, untranslated and fuzzy entries:

```
~/path/to/posplit ./zulu-gnome-glossary.po
```

This will create three files:

- zulu-gnome-glossary-translated.po all fully translated entries
- zulu-gnome-glossary-untranslated.po messages with no translation

• zulu-gnome-glossary-fuzzy.po – words that need investigation

```
rm zulu-gnome-glossary-untranslated.po
```

We discard zulu-gnome-glossary-untranslated.po since they are of no use to us.

# **Dealing with the fuzzies**

The fuzzies come in two kinds. Those that are simply wrong or needed updating and those where there was more then one translation for a given term. So if someone had translated 'File' differently across the translations we'd have an entry that was marked fuzzy with the two options displayed.

```
pofilter -t compendiumconflicts zulu-gnome-glossary-fuzzy.po zulu-gnome-glossary-

→conflicts.po
```

These compendium conflicts are what we are interested in so we use pofilter to filter them from the other fuzzies.

```
rm zulu-gnome-glossary-fuzzy.po
```

We discard the other fuzzies as they where probably wrong in the first place. You could review these but it is not recommended.

Now edit zulu-gnome-glossary-conflicts. po to resolve the conflicts. You can edit them however you like but we usually follow the format:

```
option1, option2, option3
```

You can get them into that layout by doing the following:

Of course if a word is clearly wrong, misspelled etc. then you can eliminate it. Often you will find the "problem" relates to the part of speech of the source word and that indeed there are two options depending on the context.

You now have a cleaned fuzzy file and we are ready to proceed.

## Put it back together again

```
msgcat zulu-gnome-glossary-translated.po zulu-gnome-glossary-conflicts.po > zulu-

→gnome-glossary.po
```

We now have a single file zulu-gnome-glossary.po which contains our glossary texts.

### **Create other formats**

It is probably good to make your terminology available in other formats. You can create CSV and TMX files from your PO.

```
po2csv zulu-gnome-glossary.po zulu-gnome-glossary.csv
po2tmx -l zu zulu-gnome-glossary.po zulu-gnome-glossary.tmx
```

For the terminology to be usable by Trados or Wordfast translators they need to be in the following formats:

- Trados comma delimited file source, target
- Wordfast tab delimited file source [tab] target

In that format they are now available to almost all localisers in the world.

FIXME need scripts to generate these formats.

# 1.6.7 The work has only just begun

The lists you have just created are useful in their own right. But you most likely want to keep growing them, cleaning and improving them.

You should as a first step review what you have created and fix spelling and other errors or disambiguate terms as needed.

But congratulations a Terminology list or Glossary is one of your most important assets for creating good and consistent translations and it acts as a valuable resource for both new and experienced translators when they need prompting as to how to translate a term.

# 1.6.8 Running the tools on Microsoft Windows

Since the toolkit is written in Python, it should work perfectly on Windows.

## Add the toolkit to your path

#### Windows 95/98

You might need to add the installation directory of the translate toolkit to your path

```
path "C:\Program Files\translate-toolkit\"
```

This will work for one session, but will be lost when you reboot again. Therefore you might want to add it to the autoexec.bat file.

#### Windows 2000/XP

You can add to the path permanently. Check this useful guide. You should add the following to your path:

```
C:\Programs Files\translate-toolkit\
```

If you have the Gettext tools installed, add it to your path as well:

```
C:\Program Files\GnuWin32\bin\
```

### Change Windows file to Unix file

Some programs in Windows will add CRLFs to the file which is considered rather poor practice for 110ns that require Unix files. To fix a text file, drag and drop it to the dos2unix.exe utility from http://www.bastet.com/

# 1.6.9 Cleanup translator comments

Translate Toolkit 1.1 saw source comments being converted to developer comments instead of translator comments.

This use case shows you how to get rid of the old translator comments.

### The Change

We used to put all source comments into translator comments.

```
# Some Comment
```

But now place them in developer comments.

```
#. Some Comment
```

This ensures that these source comments are updated to the newest versions from the source files, which is a good thing. Translator comments survive these updates, just like you want, while developer comments are discarded.

If you don't clean up your PO files you will now end up with:

```
# Some Comment
#. Some Comment
```

Thus a duplicated comment. Fortunately you only need to clean your PO files once.

### Removing old translator comments

**Note:** This will remove all your translator comments. So if you have some that you actually want to keep then you will need to manual editing

Removal is simple using pocommentclean:

```
pocommentclean my-po-dir
```

Which will clean all your PO files in my-po-dir

pocommentalean is simply a nice wrapper for this sed command:

```
sed -i "/^#$/d;/^#[^\:\~,\.]/d" $(find po -name "*.po")
```

This will delete all lines starting with # that are not used by PO for locations (#:), automatic/developer comments (#.), state (#,) and obsolete (#~).

You can now safely commit your changes and begin your migrations using pot2po of pomigrate2

# 1.6.10 Creating Mozilla POT files

You can do this using Mozilla source from CVS or Mercurial

## **Using Mercurial**

Since Firefox 3.1 and Thunderbird 3.0, Mozilla has switched to using Mercurial for version control. See the Mozilla's L10n on Mercurial page for instructions on how to checkout and update your Mozilla sources and 110n files.

You can use *get\_moz\_enUS.py* to extract an en-US directory from the source tree:

```
get_moz_enUS.py -s mozilla-central/ -d 110n/ -p browser
```

This will move the correct en-US files to 110n/en-US. You can now create POT files as follows:

```
moz2po -P 110n/en-US 110n/pot
```

This will create the POT files in 110n/pot using the American English files from en-US. You now have a set of POT files that you can use for translation or updating your existing PO files.

There are also *other scripts* that can help with creating and updating POT and PO files for Mozilla localisation.

### **Using CVS**

Firefox versions before 3.1 and Thunderbird versions before 3.0 still has its source in CVS. Check out files from the Mozilla repository. If you don't want to checkout all files do:

```
make -f client.mk 110n-checkout
```

The English files are in the mozilla/ module, while the translated files all reside in the llon/ module. They have different structure but not enough to kill you.

Once you have checked out mozilla/ you will need to get the correct files for en-US. To do this we will create en-US as a pseudo language.

```
make -f tools/110n/110n.mk create-en-US
```

This will move the correct en-US files to 110n/en-US. You can now create POT files as follows:

```
moz2po -P 110n/en-US 110n/pot
```

This will create the POT files in 110n/pot using the American English files from en-US. You now have a set of POT files that you can use for translation or updating your existing PO files.

### 1.6.11 Document translation

Translating documents can be quite different from translating software interfaces. Many issues specific to software localisation might not be relevant in documents, such as accelerators, translation length, constructed phrases, etc. However, document translation has several other issues that is good to be aware of.

### **Preparing for translation**

Ideally a document should be prepared for translation. A good source document will make translation easier. Possibilities:

- Proofread the document (spelling, grammar, clarity)
- Use consistent terminology
- Read "writing for translation"

• For structured documents, use proper structure like headings and subheadings instead of using style only.

#### **Translation**

A lot can be said about translation in general, but this is only meant to give you some tips.

Be to be aware of issues arising out of translation memory. You could possibly have exact matches (identical string translated before), or In Context Exact (ICE) matches, where some translation tools will specifically indicate that the translation is identical, but also that the surrounding text from the paragraph is the same. It could also indicate agreement with regards to domain, file, date, etc.

### Post-processing

After generating the translated document, you very likely need to do some post processing. Things to consider:

- Ensuring correct translation in cases where context might not have been obvious during translation
- Document layout, page layout
- · Fonts or other styling changes
- Style of generated content, such as numbers
- Generated sections, such as Table of contents, list of figures, index, variables
- Migrating an older version of your translations to the latest templates
- · Checking for technical errors in your translations
- Translating using only a spreadsheet (a look at the whole roundtrip from PO to CSV and back)
- Creating OpenOffice.org POT files
- Checking for inconsistencies in your translations
- Creating a terminology list from your existing translations
- Running the tools on Microsoft Windows
- Using phase for the complete translation roundtrip
- Cleanup translator comments
- Creating Mozilla POT files
- Document translation

# 1.7 Translation Related File Formats

These are the different storage formats for translations and files associated with translations that are supported by the toolkit. See also *Standards conformance* for standards conformance.

The Translate Toolkit implements a set of *classes* for handling translation files which allows for a uniform API which covers other issues such as *quoting and escaping* of text.

# 1.7.1 Primary translation formats

### **XLIFF**

XLIFF[\*] is the OASIS standard for translation.

#### References

- · XLIFF Standard
- OASIS XLIFF Technical Committee website

### **Flavours**

XLIFF also has documents that specify the conversion from various standard source documents and localisation formats.

- PO For conformance to the po2xliff spec, see xliff2po.
  - Draft XLIFF 1.2 Representation Guide for Gettext PO
- HTML not implemented
  - Draft XLIFF 1.2 Representation Guide for HTML
- Java (includes .properties and Java resource bundles) not implemented
  - Draft XLIFF 1.2 Representation Guide for Java Resource Bundles
- ICU Resource Bundles not officially being developed by XLIFF Proposed representation guide

## Standard conformance

## **Done**

- · File creation and parsing
- API can create multiple files in one XLIFF (some tools only read the first file)
- source-language attribute
- · trans-unit with
  - note: addnote() and getnotes()
  - state
    - \* fuzzy: isfuzzy() and markfuzzy()
    - \* translated: marktranslated()
    - \* approved
    - \* needs-review-translation: isreview(), markreviewneeded()
  - id: setid()
  - context-group: createcontextgroup()
- · context groups

· alt-trans

## XLIFF and other tools

Here is a small report on XLIFF support by Windows programs.

### **PO Files**

PO files use the file format of the Gettext tools.

#### See also:

Gettext manual

# **Supported Features**

- Headers
- Language header (since gettext version 0.17)
- Plural forms and plural form handling
- Message context

```
msgctxt "noun"
msgid "View"
msgstr ""
```

• Normal comments

```
# this is another comment
```

• Automatic comments

```
#. comment extracted from the source code
```

• Source location comments

```
#: sourcefile.xxx:35
```

• Typecomments

```
#, fuzzy
```

• Msgidcomments, also known as KDE style comments as they are used by KDE for message disambiguation and comments to translators.

**Note:** Support for this is being phased out in favor of msgctxt.

```
msgid "_: comment\n"
"translation"
```

· Obsolete messages

```
#~ msgid "Blah"
#~ msgstr "Bleeh"
```

· Previous msgid

```
#| msgid "previous message"
```

· Previous msgctxt

```
#| msgctxt "previous context"
```

## 1.7.2 Other translation formats

#### **CSV**

CSV (Comma Separated Values) is a simple file format for general data interchange. It can be used in the toolkit for simple data interchange, and can be edited with most spreadsheet programs. There is no formal specification for the CSV file format, but more information can be acquired from Comma-Separated Values

#### Conformance

CSV files were initially used to convert from and to po files, and therefore contained three columns as follows:

Col-	Description
umn	
loca-	A column with the location of the original msgid (in other words, a line in a programming source file, as
tion	indicated in the #: comments of PO files).
source	The source text (or msgid)
target	The target text (or msgstr)

Tabs and newlines are maintained, although it is not clear how easy it is to edit these things in a spreadsheet.

Quoting is a problem, because the different spreadsheet programs handle these things differently. Notably, Microsoft's excel handles single quotes slightly differently. In future, it might be worthwhile to handle excel CSV as a different format from other CSV files. An entry like 'mono' is ambiguous as it is not sure whether this refers simply to the word *mono* or to the entry 'mono' quoted with single quotes. (Example from Audacity pot file)

### **INI Files**

Also known as initialisation files. These are in some cases used to store translations.

### **Conformance**

The toolkit uses iniparse, an INI file parser that preserves layout and follows the .ini format as supported by the Python language.

### **Dialects**

The format supports two dialects:

- · default: standard iniparse handling of INI files
- inno: follows Inno escaping conventions

#### References

Further information is available on .ini files:

- Wikipedia INI file format article
- Unofficial specification

## Mozilla and Java properties files

The Translate Toolkit can manage Java .properties files with the *prop2po* and po2prop tool. As part of the Mozilla localisation process, the *moz2po* tool handles the properties files along with the other files. The tools can also handle Skype .lang files. Some related formats with their own documentation:

- Mac OSX strings
- Adobe Flex properties files.

#### **Features**

- Fully manage Java escaping (Mozilla non-escaped form is also handled)
- Preserves the layout of the original source file in the translated version

New in version 1.12.0.

• Mozilla accelerators – if a unit has an associated access key entry then these are combined into a single unit

### Not implemented

• We don't allow filtering of unchanged values. In Java you can inherit translations, if the key is missing from a file then Java will look to other files in the hierarchy to determine the translation.

# **Examples**

```
editmenu.label = "Edit"
saveas.label = "Save As"
```

### References

- Java Properties Class's load() describes the properties format.
- http://www.oracle.com/webfolder/technetwork/jsc/dtd/properties.dtd alternate XML based property representation

### **Mozilla DTD format**

Mozilla makes use of a .dtd file to store many of its translatable elements, the moz2po converter can handle these.

#### References

· XML specification

### **Features**

- Comments these are handled correctly and integrated with the unit
- Accelerators if a unit has an associated access key entry then these are combined into a single unit
- Translator directive all LOCALIZATION NOTE items such as DONT\_TRANSLATE are handled and such items are discarded
- Entities some entities such as & or " are expanded when reading DTD files and escaped when writing them, so that translator see and type & and " directly

#### Issues

We don't expand some character entities like <, &#38; – this doesn't break anything but it would be nicer to see © rather than &copy;

# OpenOffice.org GSI/SDF format

OpenOffice.org uses an internal format called SDF to manage localisation text. The toolkit can successfully manage all features of this format converting it to XLIFF or PO format with the *oo2po* and oo2xliff tools.

### **Features**

- Handles all translatable text from the SDF
- Can also use 'x-comments' 'language' found in the SDF to provide translator comments

### **PHP**

Many PHP programs make use of a localisable string array. The toolkit supports the full localisation of such files with *php2po* and po2php.

### Conformance

Our format support allows:

• Single and double quoted strings (both for keys and values)

```
<?php
$variable = 'string';
$messages["language"] = 'Language';
define('item', "another string");</pre>
```

• PHP simple variable syntax

```
<?php
$variable = 'string';
$another_variable = "another string";</pre>
```

• PHP square bracket array syntax

```
<?php
$messages['language'] = 'Language';
$messages['file'] = "File";
$messages["window"] = 'Window';
$messages["firewall"] = "Firewall";</pre>
```

· PHP array syntax

New in version 1.7.0.

```
<?php
// Can be 'array', 'Array' or 'ARRAY'.
$lang = array(
   'name' => 'value',
   'name2' => "value2",
   "key1" => 'value3',
   "key2" => "value4",
);
```

· PHP define syntax

New in version 1.10.0.

```
<?php
define('item', 'string');
define('another_item', "another string");
define("key", 'and another string');
define("another_key", "yet another string");</pre>
```

• PHP short array syntax

New in version 2.3.0.

```
<?php
$variable = [
    "foo" => "bar",
    "bar" => "foo",
];
```

• Heredoc

New in version 2.3.0.

```
<?php
$variable = <<<EOT</pre>
```

(continues on next page)

(continued from previous page)

```
bar
EOT;
```

Nowdoc

New in version 2.3.0.

```
<?php
$variable = <<<'EOD'
Example of string
spanning multiple lines
using nowdoc syntax.
EOD;</pre>
```

• Escape sequences (both for single and double quoted strings)

```
<?php
$variable = 'He said: "I\'ll be back"';
$another_variable = "First line \n second line";
$key = "\tIndented string";</pre>
```

· Multiline entries

• Various layouts of the id

```
<?php
$string['name'] = 'string';
$string[name] = 'string';
$string[ 'name' ] = 'string';</pre>
```

• Comments

Changed in version 1.10.0.

```
<?php
# Hash one-line comment
$messages['language'] = 'Language';

// Double slash one-line comment
$messages['file'] = 'File';

/*
    Multi-line
    comment

*/
$messages['help'] = 'Help';</pre>
```

• Whitespace before end delimiter

New in version 1.10.0.

```
<?php
$variable = 'string' ;

$string['name'] = 'string' ;

$lang = array(
    'name' => 'value' ,
);

define('item', 'string' );
```

· Nested arrays with any number of nesting levels

New in version 1.11.0.

· Whitespace in the array declaration

New in version 1.11.0.

```
<?php
$variable = array (
   "one" => "this",
   "two" => "that",
);
```

• Blank array declaration, then square bracket syntax to fill that array

New in version 1.12.0.

```
<?php
global $messages;
$messages = array();

$messages['language'] = 'Language';
$messages['file'] = 'File';</pre>
```

• Unnamed arrays:

New in version 2.2.0.

```
<?php
return array(
    "one" => "this",
);
```

• Array entries without ending comma:

New in version 2.3.0.

```
<?php
$variable = array(
    "one" => "this",
    "two" => "that"
);
```

• Array entries with space before comma:

New in version 2.3.0.

• Nested arrays declared on the next line:

New in version 2.3.0.

• Nested arrays with blank entries:

New in version 2.3.0.

• Strings with slash asterisk on them:

New in version 2.3.0.

```
<?php
$variable = array(
    'foo' => 'Other value /* continued',
);
```

• Array entries with value on next line:

New in version 2.3.0.

```
<?php
$variable = array(
    'foo' =>
         'bar',
);
```

• Array defined in a single line:

New in version 2.3.0.

· Keyless arrays:

New in version 2.3.0.

```
<?php
$days = array('Sunday', 'Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday',
→'Saturday');</pre>
```

• Nested arrays without key for a nested array:

New in version 2.3.0.

```
<?php
$lang = array(array("key" => "value"));
```

• Concatenation of strings and variables:

New in version 2.3.0.

```
<?php
$messages['welcome'] = 'Welcome ' . $name . '!';
$messages['greeting'] = 'Hi ' . $name;</pre>
```

• Assignment in the same line a multiline comment ends:

New in version 2.3.0.

```
<?php
/*
    Multi-line
    comment
*/ $messages['help'] = 'Help';</pre>
```

### **Non-Conformance**

The following are not yet supported:

• Keyless arrays assigned to another array:

```
<?php
$messages['days_short'] = array('Sun', 'Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat');</pre>
```

### Qt.ts

The Qt toolkit uses a .ts file format to store translations which are traditionally edited using Qt Linguist.

#### References

The format is XML and seems to only have been documented properly since Qt 4.3

- Current DTD Specification for Qt 5, older versions; Qt 4.3
- http://svn.ez.no/svn/ezcomponents/trunk/Translation/docs/linguist-format.txt

## **Complete**

Note that *ts2po* uses and older version and does not support all of these features. Virtaal, Pootle and other users of the new ts class support the following:

- Context
- Message: status (unfinished, finished, obsolete), source, translation, location
- Notes: comment, extracomment, translatorcomment (last two since Toolkit 1.6.0)
- Plurals: numerusform

### **TODO**

**Note:** A new parser has been added to the toolkit in v1.2. This allows Virtaal, *pocount* and other users to work with v1.1 of the .ts format. This corrects almost all of the issues listed below. The converter *ts2po* continues to use the older storage class and thus continue to experience some of these problems.

- Compliance with above DTD
- byte: within various text areas
- translation: obsolete (currently handled with comments in conversion to PO. But should be able to convert Obsolete PO back into obsolete TS. This might mean moving this format properly onto the base class).
- · lengthvariants
- \*comment: various new comment fields
- old\*: ability to store previous source and comments

### **Validate**

These might work but need validation

• Encoding handling for non-UTF-8 file encodings

## Windows RC files

New in version 1.2.

Windows .rc files, or resource files, are used to store translatable text, dialogs, menu, etc. for Windows applications. The format can be handled by the Translate Toolkit rc2po and po2rc.

## Conformance

The actual specification of .rc files is hard to come by. The parser was built using WINE .rc files as a reference. This was done as WINE is a good target for .rc translations. We are confident though that the extraction will prove robust for all .rc files.

## **Useful resource**

- · RC converter
- ReactOS translation instructions

## Supported elements

- DIALOG, DIALOGEX: All translatables
- MENU: POPUP, MENUITEM
- STRINGTABLE
- LANGUAGE: We only parse the first language tag, further LANGUAGE section are ignored

# **Bugs**

- There may be problems with very deeply nested MENU's
- LANGUAGE elements cannot yet be updated in po2rc (Issue 360)

## **Mac OSX strings**

New in version 1.8.

Mac OSX .strings files are used for some Cocoa / Carbon application localization, such as for the iPhone, iPod, and OSX. They are somewhat similar to Java properties, and therefore *prop2po* and po2prop are used for conversion.

## References

- Localising string resources
- Manual creation of .strings files
- String format specifiers

# **Adobe Flex properties files**

New in version 1.8.

Adobe Flex applications use *Java properties* encoded in UTF-8. The *prop2po* and po2prop commands are used for conversion.

#### References

• Description for Adobe Flex properties files

## Haiku catkeys

New in version 1.8.

Localisation for the Haiku operating system is done with a file format called catkeys. It is a bilingual file format.

## Links

- · Some notes about the format
- Some example files

## **Android string resources**

Android programs make use of localisable string resources.

Note: The toolkit supports this format, but still doesn't provide any converter.

#### References

- Android Resource files reference
- · Android String resources reference
- · Localizing Android Applications tutorial
- Reference for translatable attribute

## .NET Resource files (.resx)

.Net Resource (.resx) files are a monolingual file format used in Microsoft .Net Applications. The .resx resource file format consists of XML entries, which specify objects and strings inside XML tags. It contains a standard set of header information, which describes the format of the resource entries and specifies the versioning information for the XML used to parse the data. Following the header information, each entry is described as a name/value pair.

Comments can be added per string using the optional <comment> field. As only one comment field is available, both translator and developer comments are stored in the same place. Translator comments are automatically wrapped with brackets and pre-fixed with 'Translator Comment:' during the po2resx process to make it easy to distinguish comment origin inside the .resx files.

Example:

resx2po and po2resx are used for conversion.

#### References

- · Resources in .Resx File Format
- ASP.NET Web Page Resources Overview

## Mozilla .lang files

Mozilla's custom .lang format is used for some of their websites.

#### References

- · .lang specification
- www.mozilla.org repository of translations
- CSV
- INI Files (including Inno Setup .isl dialect)
- Java Mozilla and Java properties files (also Mozilla derived properties files)
- Mozilla Mozilla DTD format
- OpenOffice.org *OpenOffice.org GSI/SDF format* (Also called SDF)
- PHP translation arrays
- Qt Linguist *Qt* .ts (both 1.0 and 1.1 supported, 1.0 has a converter)
- · Symbian localization files
- Windows Windows RC files files
- Mac OSX Mac OSX strings files (also used on the iPhone) (from version 1.8)
- Adobe Adobe Flex properties files (from version 1.8)
- Haiku Haiku catkeys (from version 1.8)
- Android string resources (supports storage, not conversion)
- .NET Resource files (.resx) .NET Resource files (.resx)
- Mozilla .lang files

# 1.7.3 Translation Memory formats

#### **TMX**

TMX is the LISA OSCAR standard for translation memories.

## Standard conformance

Summary: TMX version 1.4 conformance to Level 1, except that no markup is stripped.

- All required header fields are supplied.
- The adminlang field in the header is always English.
- None of the optional header fields are supplied.
- We assume that only two languages are used (source and single target language).
- No special consideration for segmentation.
- Currently text is treated as plain text, in other words no markup like HTML inside messages are stripped or interpreted as it should be for complete Level 1 conformance.

# **Wordfast Translation Memory**

The Wordfast format, as used by the Wordfast translation tool, is a simple tab delimited file.

The storage format can read and write Wordfast TM files.

#### Conformance

- Escaping The format correctly handles Wordfast & 'XX; escaping and will unescape and escape seamlessly.
- Soft-breaks these are not managed and are left as escaped
- Replaceables these are not managed
- Header Only basic updating and reading of the header is implemented
- Tab-separated value (TSV) the format correctly handles the TSV format used by Wordfast. There is no quoting, Windows newlines are used and the \tau is used as a delimiter (see issue 472)
- TMX
- Wordfast Translation Memory: TM
- Trados: .txt TM (from v1.9.0 read only)

# 1.7.4 Glossary formats

#### OmegaT glossary

New in version 1.5.1.

OmegaT allows a translator to create a terminology list of glossary files. It uses this file to provide its glossary matches to the OmegaT users.

## Format specifications

The glossary files is a tab delimited files with three columns:

- 1. source
- 2. target

#### 3. comment

The files is stored in the system locale if the files extension is .txt or in UTF-8 if the file extension is .utf8.

#### Conformance

The implementation can load files in UTF-8 or the system encoding.

#### **Issues**

- There has not been extensive testing on system encoded files and there are likely to be issues in these files for encodings that fall outside of common ASCII characters.
- Files with additional columns are read correctly but cannot be written.

# Qt Phrase Book (.qph)

New in version 1.2.

Qt Linguist allows a translator to collect common phrases into a phrase book. This plays a role of glossary lookup as opposed to translation memory.

## Conformance

There is no formal definition of the format, although it follows a simple structure

## Missing features

There are no missing features in our support in the toolkit. The only slight difference are:

- · We don't focus on adding and removing items, just updating and reading
- Comments are not properly escaped on reading, they might be on writing
- An XML header is output on writing while it seems that no files in the wild contain an XML header.
- The <definition> is aimed at users, the toolkits addnote feature focuses on programmer, translators, etc comments while there is really only one source of comments in a .qph. This causes duplication on the offline editor.

## **TBX**

TBX is the LISA OSCAR standard for terminology and term exchange.

For information on more file formats, see Standards conformance.

## References

- Standard home page
- · Specification
- ISO 30042 TBX is an approved ISO standard
- Additional TBX resources

You might also be interested in reading about TBX-Basic – a simpler, reduced version of TBX with most of the useful features included.

Additionally notes and examples about TBX are available in Terminator TBX conformance notes which might help understanding this format.

Also you might want to use TBXChecker in order to check that TBX files are valid. Check the TBXChecker explanation.

#### Conformance

Translate Toolkit TBX format support allows:

- · Basic TBX file creation
- Creating a bilingual TBX from CSV using csv2tbx
- Using <tig> tags only
- Simple extraction of Parts of Speech and definitions

# **Non-Conformance**

The following are not yet supported:

- id attribute for <termEntry> tags
- Multiple languages
- Multiple translations in the same language
- · Cross references
- Context
- Abbreviations
- Synonyms
- <ntig> tag, read and write

Other features can be picked from the Terminator TBX conformance notes which also include examples and notes about the TBX format.

## Universal Terminology eXchange (UTX)

New in version 1.9.

UTX is implemented by the Asia-Pacific Association for Machine Translation

## Resources

- · UTX site
- Current Specification (implementation is based on UTX 1.0 which is no longer available)

#### Conformance

The Translate Toolkit implementation of UTX can correctly:

- Handle the header. Although we don't generate the header at the moment
- Read any of the standard columns and optional columns. Although we can access these extra columns we don't do much with them.

Adjustments and not implemented features where the spec is not clear:

- We do not implement the "#." comment as we need clarity on this
- The "<space>" override for no part of speech is not implemented
- The spec calls for 2 header lines, while examples in the field have 2-3 lines. We can read as many as supplied but assume the last header line is the column titles
- We remove # from all field line entries, some examples in the field have #tgt as a column name
- *OmegaT glossary* (from v1.5.1)
- Qt Phrase Book (.qph)
- TBX
- Universal Terminology eXchange (UTX) (from v1.9.0)

## 1.7.5 Formats of translatable documents

#### HTML

The Translate Toolkit is able to process HTML files using the html2po converter.

The HTML support is basic, so please be aware of that.

#### Conformance

- Can identify almost all tags and attributes that are localisable.
- Does not convert HTML entities (e.g. ©) to normal strings
- It does not handle inline elements well and will drop them, so complicated HTML might not make it through the filter

#### References

• Using character entities: http://www.w3.org/International/questions/qa-escapes

#### Flat XML

The Translate Toolkit is able to process flat XML files using the *flatxml2po* converter.

Flat XML (eXtensible Markup Language) is a simple monolingual file format similar to a very basic form of the *Android string resources* format. Flat in this context means a single level of elements wrapped in the root-element with no other structuring.

#### Conformance

• Single-level XML with attributes identifying a resource:

```
<root>
    <str key="hello_world">Hello World!</str>
    <str key="resource_key">Translated value.</str>
</root>
```

• Customizable element- and attribute-names (including namespaces):

```
<dictionary xmlns="urn:translate-toolkit:flat-xml-dictionary">
    <entry name="hello_world">Hello World!</entry>
    <entry name="resource_key">Translated value.</entry>
</dictionary>
```

Value whitespace is assumed to be significant (equivalent to setting xml:space="preserve"):

```
<root>
     <str key="multiline">The format assumes xml:space="preserve".
There is no need to specify it explicitly.

This assumption only applies to the value element; not the root element.

</root>
```

• Non-resource elements and attributes are preserved (assuming the same file is also used when converting back to XML):

```
<root>
    <str key="translate_me">This needs to be translated</str>
    <const name="the_answer" hint="this isn't translated">42</const>
    <str key="important" priority="100">Some important string</str>
</root>
```

• Indentation can be customized to match an existing and consistent style:

**Note:** To avoid potential issues and extraneous changes in diffs, this format always forces an ending linefeed by default for compatibility with various Version control systems (such as Git).

## **Non-Conformance**

While the format is flexible, not all features are supported:

- Mixed element/attribute names (as well as different namespaces for root- and value-element) and nested structures additional child elements. This format intentionally focuses on a simple structure that can be used by other languages (such as XSLT).
- Comments are preserved on roundtrips, but are not carried over into the resulting PO Files.
- XML Fragments and non-wellformed XML.

#### References

XML specification

#### **iCalendar**

Support for iCalendar (\*.ics) files. This allows calendars to be localised.

The format extracts certain properties from VEVENT objects. The properties are limited to textual entries that would need to be localised, it does not include entries such as dates and durations that would indeed change for various locales.

#### Resources

- rfc2445 Internet Calendaring and Scheduling Core Object Specification (iCalendar)
- iCal spec in a simple adaptation of the rfc that makes it easy to refer to all sections, items and attributes.
- VObject the python library used to read the iCal file.
- · iCalender validator
- iCalendar
- Components and their properties

#### Conformance

We are not creating iCal files, simply extracting localisable information and rebuilding the file. We rely on VObject to ensure correctness.

The following data is extracted:

- VEVENT:
  - SUMMARY
  - DESCRIPTION
  - LOCATION
  - COMMENTS

No other sections are extracted.

#### **Notes**

## LANGUAGE: not a multilingual solution

It is possible to set the language attribute on an entry e.g.:

```
SUMMARY:LANGUAGE=af;New Year's Day
```

However since only one SUMMARY entry is allowed this does not allow you to specify multiple entries which would allow a single multilingual file. With that in mind it is not clear why the LANGUAGE attribute is allowed, the examples they give are for LOCATION entries but that is still not clearly useful.

## **Development Notes**

If we use LANGUAGE at all it will be to ensure that we specify that an entry is in a given language.

#### **JSON**

New in version 1.9.0.

JSON is a common format for web data interchange.

Example:

```
"firstName": "John",
"lastName": "Smith",
"age": 25,
"address": {
    "streetAddress": "21 2nd Street",
    "city": "New York",
    "state": "NY",
    "postalCode": 10021
"phoneNumbers": [
    {
        "type": "home",
        "number": "212 555-1234"
    },
    {
        "type": "fax",
        "number": "646 555-4567"
    }
]
```

## **YAML**

New in version 2.0.0.

YAML is a common format for web data interchange. The two variants of YAML files are supported:

- Plain YAML files.
- Ruby YAML localization files with root node as language. This variant supports plurals as well.

## **Non-Conformance**

The following are not yet supported (in most cases these are properly parsed, but not saved in round trip):

• Multiline strings:

```
include_newlines: |
    exactly as you see
    will appear these three
    lines of poetry
```

· Abbreviated lists:

```
first-name: [Sun, Mon, Tue, Wed, Thu, Fri, Sat]
```

• Abbreviated dictionaries:

```
martin: {name: Martin D'vloper, job: Developer, skill: Elite}
```

• Quoted strings:

```
foo: "quote, double"
bar: 'quote, single'
```

· Escaped quotes:

```
foo: "Hello \"World\"."
bar: 'Hello \'World\'.'
```

Avoid escaping quotes:

```
spamm: 'avoid escaping "double quote"'
eggs: "avoid escaping 'single quote'"
```

• Newlines:

```
foo: "Hello \n World."
```

· Booleans:

```
foo: True
```

## **OpenDocument Format**

This page summarises the support for the OpenDocument format (ODF) in the Translate Toolkit. This currently involves only the *odf2xliff* and xliff2odf converters.

The Translate Toolkit aims to support version 1.1 of the ODF standard, although it should work reasonably well with older or newer files to the extent that they are similar.

Our support is implemented to classify tags as not containing translatable text, or as being inline tags inside translatable tags. This approach means that new fields added in future versions will automatically be seen as translatable and should still be extracted successfully, even if the currently released versions of the Translate Toolkit are not aware of their existence.

· Currently used and classified tags

More complex tag uses are still needed to extract 100% correctly in some complex cases. For more information, see the list of issues from testing.

## **Simple Text Documents**

The Translate Toolkit can process simple Text files. This is very useful for translating installation files and READMEs. The processing of these files is performed by the *txt2po* converter.

In some cases you will need to adjust the source text for the conversion management to work well. This is because the text file format support considered units to be space separated blocks of text.

## **Example**

```
Heading
======

Paragraph One

Paragraph Two:
* First bullet
* Second bullet
```

This example will result in three units. The first will include the underline in the header. The third will include all the bullet points in one paragraph together with the paragraph lead in.

# Wiki Syntax

The Translate Toolkit can manage Wiki syntax pages. This is implemented as part of the *text* format and the conversion is supported in *txt2po*.

Those who edit wikis will appreciate that wiki text is simply a normal text document edited using a form of wiki syntax. Whether the final storage is a database or a flat file the part that a user edits is a simple text file.

The format does not support all features of the wiki syntax and will simply dump the full form if it doesn't understand the text. But structures such as headers and lists are understood and the filter can remove these are correctly add them.

## **Supported Wiki Formats**

The following is a list of the wikis supported together with a list of the items that we can process:

- dokuwiki heading, bullet, numbered list
- MediaWiki heading, bullet, numbered list

#### Possible uses

As part of a localisation process for a wiki this format and the filters could be used to provide a good localisation of existing wiki content.

With further enhancement the tool could probably be capable of converting from one wiki syntax to another, but that is of course not its main aim

#### Additional notes on MediaWiki

Media wiki can also export in XML format, see http://en.wikipedia.org/wiki/Special:Export and http://www.mediawiki.org/wiki/Manual:Parameters\_to\_Special:Export this however exports in XML so not directly usable by txt2po.

For importing please see http://en.wikipedia.org/wiki/Help:Import this is disabled on most wikis so not directly usable currently.

## **Subtitles**

New in version 1.4.

The translation of subtitles are supported in the toolkit with the commands *sub2po* and po2sub.

The following formats are supported for subtitles:

- MicroDVD
- MPL2
- MPsub
- SubRip (.srt)
- SubViewer 2.0 (.sub)
- TMPlayer
- · Sub Station Alpha
- · Advanced Sub Station Alpha

YouTube supports a number of formats

## Implementation details

Format support is provided by Gaupol a subtitling tool. Further enhancement of format support in Gaupol will directly benefit our conversion ability.

## **Usage**

It must be noted that our tools provide the ability to localise the subtitles. This in itself is useful and allows a translator to use their existing localisation tools. But this is pure localisation and users should be aware that they might still need to post edit their work to account for timing, limited text space, limits in the ability of viewers to keep up with the text.

For most cases simply localising will be good enough. But in some cases the translated work might need to be reviewed to fix any such issues. You can use Gaupol to perform those reviews.

- HTML
- Flat XML (single-level XML)
- iCalendar
- JSON
- YAML
- OpenDocument all ODF file types

- Text plain text with blocks separated by whitespace
- Wiki DokuWiki and MediaWiki supported
- Subtitles various formats (v1.4)

## 1.7.6 Machine readable formats

#### Gettext .mo

The Gettext .mo (Machine Object) file is a compiled *Gettext PO* file. In execution Gettext enabled programs retrieve translations from the .mo file. The file contains arrays for the English and the translations, an optional hash can speed up the access to the translations.

#### Conformance

The toolkit can create .mo files from PO or XLIFF files, handling plurals and msgctxt. It can also read .mo files, allowing counting, etc and also allowing the .mo files to act as a translation memory.

Changed in version 1.2: The hash table is also generated (the Gettext .mo files works fine without it). Due to slight differences in the construction of the hashing, the generated files are not identical to those generated by msgfmt, but they should be functionally equivalent and 100% usable. Issue 326 tracked the implementation of the hashing. The hash is platform dependent.

## Qt.qm

A .qm file is a compiled *Qt linguist* file. In many ways it is similar to Gettext, in that it uses a hashing table to lookup the translated text. In older version they store only the hash and the translation which doesn't make the format useful for recovering translated text.

#### Conformance

The toolkit can read .qm files correctly. There are some unimplemented aspects of the message block, but these seem to be legacy features and require examples to be able to implement the feature.

The .qm implementation cannot write a .qm file, thus you are only able to use this format in a read-only context: counting messages (*pocount*), reading in messages for a TM or using it as a source format for a converter e.g. a possible qm2xliff converter.

#### **TODO**

- Writing
  - Hash algorithm
- Gettext Gettext .mo
- Qt Qt.qm (read-only)

# 1.7.7 In development

# 1.7.8 Unsupported formats

Formats that we would like to support but don't currently support:

#### Mozilla 120n

L20n is a proposed method for localisation in the Mozilla project. https://wiki.mozilla.org/L20n

#### Comments

Here we collect some comments about the possibility of 120n support in our tools.

Our tools are based on the idea of units. Each unit has an associated (source, target) pair, possibly containing more, like comments, state, etc. We prefer to build our richer tools against bilingual formats like *PO Files*, *XLIFF*, *Qt .ts*, etc. and provide converters to these formats when we want to support guide/monolingual formats. Pootle also implements its own bilingual format in its database.

120n is based on the idea of entity soup, or object soup. Files are (at least in the first iteration) monolingual, and each language defines a set of objects which can contain one or more presentation forms of an entity/string, to vary it according to gender, case, declension, plural, time of day, etc. A language can define extra entities to help in constructing others by "factoring out" things, if you will. The structure of each object is left to the programmer for the target language to define.

# **Findings**

Our tools work on the principle of units. A unit being in its simplest form a source to target mapping, in other words the English source text plus the target translation make a unit. Our code has, for a long time now, a good understanding of units that don't have a one to one mapping. In PO those are plural units. This is where N source strings map to M target strings. We currently only have 1-M and 2-M mappings for Qt and PO files. 120n introduces N-M mappings which we don't currently support.

120n is of course working around an idea of translation objects not strings. But I think the string metaphor works in most cases to ease explanation.

The closest thing we have to this is plural support by means of multistrings.

120n is pretty powerful with the ability to arbitrarily make up functions/macros that then map to the correct string to use in the translation. In PO the number of possible (plural) strings is mapped before you begin, 120n potentially has any arbitrary mapping.

We could write a simple converter to another format for simple string based objects without further structure, but that doesn't expose the power of 120n yet, and doesn't handle complexity in the source text if it was present.

#### Issues

- N-M mapping. We need to support arbitrary mappings between source and target
- Determining N and M on the fly. We need to have the ability to determine N and M in real time. So that would mean being able to read 120n files and determine what function is used, then determine how many possible results that function can return. We'd do that for both source and target. Thus we'd get N and M counts which we can use in the interface.

- GUI for Pootle/Virtaal to allow dynamic source and target numbers. We already adapt to N and M on both platforms, but doing this on the fly is harder.
  - If we assume that functions are implemented once in a common library and named the same (for example for plural support) then this is easy.
  - If functions are arbitrarily implemented per target file but at least named the same then this is harder.
  - If names of functions are changed then we'd need to present the ability to change the function that a
    translator would use in their translation. How to do this so that it isn't confusing would require quite some
    thought.
  - Being able to write functions on the fly within the translation tool would most likely be the ultimate ability.
     We suspect we won't need to address that level just yet.
- Backend file store. We have two options.
  - Covert to a bilingual store this is what we do in moz2po.
  - Support monolingual stores we can do that in Pootle, but it needs wider testing. In Virtaal we do automatic conversion to bilingual formats, but is currently disabled. To enable this so that we can rely on it we'd need some work on both Pootle and Virtaal; in testing and in managing source and target files changes reliably.
- More complex 120n interactions. These start pushing the translation tool into an IDE but would include:
  - A translator making a 1-1 into a 1-M (to add gender, vary on the time of day, platform, etc). Since no functions are present in the source we'd need to have access to a library of functions or have a structured object editor.
  - A translator might want to define a local entity (an entity which is not in the original source document), or it might be there from before.
- Still needs some thought on how to do anything meaningful with our current translation features like TM, MT, quality checks.

## **Approach**

The problems above really highlight the approach we'd take to implement 120n in our tools.

- Expand the toolkit to do N-M mapping
- Include 120n parser to allows N and M mapping determination on the fly
- Convert to an interim store. Before tackling the monoligual side we'd look at converting to an interim store to reduce the risks. We'd determine what to use at the time. The only thing certain is that it would not be PO, as PO can't do N source strings.
- GUI changes. This would be to allow N and M to change dynamically. But we'd limit this to at first relying on 1-1 mappings of functions. Thus plural() in source means plural() in target.

At this point we have a usable translation tool for 120n. The next steps would be about making that support more robust. Each of these would really be determined closer to the time.

- Adaptable N and M. First allowing functions within a file to adapt the values of N and M.
- GUI selection of functions. Ability to select functions from within the GUI.
- Monolingual on the fly. We'd then look at the monolingual side of things. This would be so that we can work on the source and target without the need for the interim store.

We're now really at a position where we've solved things up to point 4 above. Addressing issues in point 5 and 6 would be the next steps.

## Wireless Markup Language

This page documents the support for WML and is used for planning our work on it.

This is implemented as a generic XML document type that is handled similarly to the way the developers/projects/odf project handles ODF documents.

- · Wordfast:
  - Glossary tab-delimited "source, target, comment" i.e. like OmegaT but unsure if any extension is required.
- Apple:
  - AppleGlot
  - .plist see issue 633 and plistlib for Python
- Adobe:
  - FrameMaker's Maker Interchange Format MIF (See also python-gendoc, and Perl MIF module)
  - FrameMaker's Maker Markup Language (MML)
- · Microsoft
  - Word, Excel, etc (probably through usage of OpenOffice.org)
  - OOXML (at least at the text level we don't have to deal with much of the mess inside OOXML). See also:
     Open XML SDK v1
  - Rich Text Format (RTF) see also pyrtf-ng
  - Open XML Paper Specification
- · XML related
  - Generic XML
  - DocBook (can be handled by KDE's xml2pot)
  - SVG
- DITA
- PDF see spec, PDFedit
- LaTeX see plasTeX, a Python framework for processing LaTeX documents
- unoconv Python bindings to OpenOffice.org UNO which could allow manipulation of all formats understood by OpenOffice.org.
- Trados:
  - TTX (Reverse Engineered DTD, other discussion)
  - Multiterm XML TSV to MiltiTerm conversion script or XLST
  - tmw
  - .txt (You can interchange using TMX) Format explanation with some examples.
- Tcl: .msg files. Good documentation
- Installers:
  - NSIS installer: Existing C++ implementation
  - WiX MSI (Microsoft Installer) creator. Localization instructions, more notes on localisation. This is a custom XML format, another one!

- catgets/gencat: precedes gettext, looking in man packages is the best information I could find. Also LSB requires it. There is some info about the source (msgfile) format on GNU website
- Wireless Markup Language
- GlossML
- Deja Vu External View: Instructions sent to a translator, Description of external view options and process
- Mozilla's 120n.

# 1.7.9 Unlikely to be supported

These formats are either: too difficult to implement, undocumented, can be processed using some intermediate format or used by too few people to justify the effort. Or some combination or these issues.

#### Standards conformance

This page links to pages documenting standard conformance for different standards or file formats.

## LISA and OASIS standards

- TMX
- XLIFF
- TBX

## Other formats

- Gettext PO
- Gettext .mo
- CSV
- Qt Linguist
- Qt .qph and .qm files
- Wordfast translation memory
- · OmegaT glossary

# Searching and matching

• Levenshtein distance

# Base classes

NOTE: This page is mostly useful for developers as it describes some programming detail of the toolkit.

For the implementation of the different storage classes that the toolkit supports, we want to define a set of base classes to form a common API for all formats. This will simplify implementation of new storage formats, and enable easy integration into external tools, such as Pootle. It will also mean less duplication of code in similar storage formats.

These ideas explained here should be seen as drafts only.

# Requirements

The base classes should be rich enough in functionality to enable users of the base classes to have access to all or most of the features that are available in the formats. In particular, the following are considered requirements:

- Seamless and hidden handling of escaping, quoting and character sets
- Parsing a file when given a file name or file contents (whole file in a string)
- Writing a file to disk
- Getting and setting source and target languages
- Accessing units, and determining if they are translatable, translated, a unique identifier for the unit in the file, etc.
- Support for plural units that can vary between different languages (as the PO format allows with msgid\_plural, etc.)

#### Other possibilities:

- Support for variable number of languages in the format. Examples: .txt and .properties support one language, PO supports two, *TMX* supports many.
- Support for "multifiles", in other words a file that contain other entities that corresponds to files in other formats. Examples: ZIP and *XLIFF*. In reality this is only used by some of the converters. This isn't present in the base class yet.

All these do not mean that all formats must support all these features, but in the formats that do support these features, it must be accessible through the base class, and it must be possible to interrogate the storage format through the base class to know which features it supports.

# The classes

A file contains a number of translation units, and possibly a header. Each translation unit contains one or more strings corresponding to each of the languages represented in that unit.

## Message/string (multistring)

This class represents a single conceptual string in a single language. It must know its own requirements for escaping and implement it internally. Escaped versions are only used for internal representation and only exposed for file creation and testing (unit tests, for example).

Note that when storing different plural forms of the same string, they should be stored in this class. The main object is the singular string, and all of the string forms can be accessed in a list at x.strings. Most of the time the object can be dealt with as a single string, only when it is necessary to deal with plural forms do the extra strings have to be taken into account.

Any string from a plural unit must be a multistring.

#### **Translation unit**

This class represents a unit of one or several related messages/strings. In most formats the contained strings will be translations of some original message/string. It must associate a language value with each message/string. It must

know how to join all contained messages/strings to compile a valid representation. For formats that support at least two languages, the first two languages will serve as "source" and "target" languages for the common case of translating from one language into another language.

Some future ideas:

As the number of languages can be seen as one "dimension" of the translation unit, plurality can be seen as a second dimension. A format can thus be classified according to the dimensionality that it supports, as follows:

- .properties files supports one language and no concept of plurals. This include most document types, such as .txt, HTML and OpenDocument formats.
- Old style PO files supported two languages and no plurals.
- New style PO files support two languages and any number of plurals as required by the target language. The plural forms are stored in the original or target strings, as extra forms of the string (See message/string class above).
- TMX files support any number of languages, but has no concept of plurality.

Comments/notes are supported in this class. Quality or status information (fuzzy, last-changed-by) should be stored. TODO: see if this should be on unit level or language level.

#### **Store**

This class represents a whole collection of translation units, usually stored in a single file. It supports the concept of a header, and possibly comments at file level. A file will not necessarily be contained alone in single file on disc. See "multifile" below.

#### Multifile

This abstraction is only used by a few converters.

This class represents a storage format that contains other files or file like objects. Examples include ZIP, XLIFF, and OpenOffice SDF files. It must give access to the contained files, and possibly give access to the translation units contained in those files, as if they are contained natively.

#### **Additional Notes**

Dwayne and I (Andreas) discussed cleaning up the storage base class. A lot of what we discussed is related to the above. A quick summary:

- Implement a new base class.
  - Flesh out the API, clean and clear definitions.
  - Document the API.
- We need to discuss the class hierarchy, e.g.:

```
base
-- po
-- text
-- xml
-- xhtml
-- lisa
-- xliff
```

(continues on next page)

(continued from previous page)

```
-- tmx
-- tbx
```

- Clean up converters.
  - Parsing of file content needs to happen only in the storage implementation of each filetype/storage type.
     Currently parsing happens all over the place.
  - Currently there are separate conversion programs for each type and direction to convert to, e.g. po2xliff and xliff2po (24 commands with lots of duplicate code in them). Ideally conversion should be as simple as:

Note that the xliffstore is being instantiated using the postore object. This works because all the data in any translation store object is accessible via the same well-defined base API. A concept class implementing the above code snippet has already been written.

- Move certain options into their respective storage classes.
  - e.g. the --duplicates option can move into po.py
- Store the meta data for a storage object.
  - Can be implemented as separate sqlite file that accompanies the real file.
  - Features not directly supported by a file format can be stored in the metadata file.
- A storage object should know all information pertaining to itself.
  - e.g. "am I monolingual?"
- We should discuss how to make an object aware that it is monolingual, bilingual or multilingual.
  - Maybe through mixin-classes?
  - How will the behaviour of a monolingual store differ from a bilingual store?

# **Quoting and Escaping**

Different translation *formats* handle quoting and escaping strings differently. This is meant to be a common page which outlines the differences

## **PO** format

Strings are quoted using double quotes. For long strings multiline quotes are done by opening and closing the quotes on each line. Usually in this case the first line is left blank. The splitting of strings over lines is transparent i.e. it does not imply line breaks in the translated strings.

Escaping is done with a backslash. An escaped double quote (") corresponds to a double quote in the original string.  $\n$  for newline,  $\t$  for tabs etc are used. Backslashes can be escaped to give a native backslash.

See also escaping in the translation guide.

## Example:

```
msgid ""
"This is a long string with a \n newline, a \" double quote, and a \\ backslash."
"There is no space between the . at the end of the last sentence "
"and the T at the beginning of this one."
```

#### **DTD** format

Strings are quoted using either double or single quotes. The quoting character may not occur within the string. There is no provision for escaping. XML entities can be used e.g. ' can be used to denote a single quote within the single-quoted string.

Some DTD files seem to have backslash-escapes, but these are anomalies: see discussion thread on Mozilla 110n-dev

## Mozilla properties format

Note that this section does not describe the Java properties files, even though they are quite similar.

It seems that the literal string  $\n$  (a backslash followed by the character 'n') and  $\t$  and  $\t$  cannot be encoded in properties files. This is the assumption of the toolkit.

# Developer's Guide

If you are a developer interested in using the Translate Toolkit for building new tools, make sure to read through this part.

# 2.1 Translate Styleguide

The Translate styleguide is the styleguide for all Translate projects, including Translate Toolkit, Pootle, Virtaal and others. Patches are required to follow these guidelines.

This Styleguide follows PEP 8 with some clarifications. It is based almost verbatim on the Flask Styleguide.

# **2.1.1 Python**

These are the Translate conventions for Python coding style.

## General

## Indentation

4 real spaces, no tabs. Exceptions: modules that have been copied into the source that don't follow this guideline.

## Maximum line length

79 characters with a soft limit for 84 if absolutely necessary. Try to avoid too nested code by cleverly placing *break*, *continue* and *return* statements.

# **Continuing long statements**

To continue a statement you can use backslashes (preceded by a space) in which case you should align the next line with the last dot or equal sign, or indent four spaces:

If you break in a statement with parentheses or braces, align to the braces:

```
this_is_a_very_long(function_call, 'with many parameters',
23, 42, 'and even more')
```

If you need to break long strings, on function calls or when assigning to variables, try to use implicit string continua-

For lists or tuples with many items, break immediately after the opening brace:

```
items = [
   'this is the first', 'set of items', 'with more items',
   'to come in this line', 'like this'
]
```

#### **Blank lines**

Top level functions and classes are separated by two lines, everything else by one. Do not use too many blank lines to separate logical segments in code. Example:

```
def hello(name):
    print('Hello %s!' % name)

def goodbye(name):
    print('See you %s.' % name)

class MyClass(object):
    """This is a simple docstring"""
```

(continues on next page)

(continued from previous page)

```
def __init__(self, name):
    self.name = name

@property
def annoying_name(self):
    return self.name.upper() + '!!!!111'
```

## **Strings**

• Double quotes are suggested over single quotes, but always try to respect the surrounding coding style. This is overruled by escaping which you should always try to avoid.

```
# Good.
str1 = "Sauron's eye"
str2 = 'Its name is "Virtaal".'

# Bad.
str3 = 'Sauron\'s eye'
str4 = "Its name is \"Virtaal\"."
```

## String formatting

While str.format() is more powerful than %-formatting, the latter has been the canonical way of formatting strings in Python for a long time and the Python core team has shown no desire to settle on one syntax over the other. For simple, serial positional cases (non-translatable strings), the old "%s" way of formatting is preferred. For anything more complex, including translatable strings, str.format is preferred as it is significantly more powerful and often cleaner.

```
# Good
print("Hello, {thing}".format(thing="world"))
print("Hello, {}".format("world"))
print("%s=%r" % ("hello", "world")) # non-translatable strings

# Bad
print("%s, %s" % ("Hello", "world")) # Translatable string.
print("Hello, %(thing)s" % {"thing": "world"}) # Use {thing}.
```

# **Imports**

Like in PEP 8, but:

- Imports should be grouped in the following order:
  - 1) \_\_future\_\_ library imports
  - 2) Python standard library imports
  - 3) Third party libraries imports
  - 4) Translate Toolkit imports
  - 5) Current package imports, using explicit relative imports (See PEP 328)

- A blank line must be present between each group of imports (like in PEP8).
- Imports on each group must be arranged alphabetically by module name:
  - Shortest module names must be before longer ones: from django.db import ... before from django.db.models import ....
- import ... calls must precede from ... import ones on each group:
  - On each of these subgroups the entries should be alphabetically arranged.
  - No blank lines between subgroups.
- On from ... import
  - Use a CONSTANT, Class, function order, where the constants, classes and functions are in alphabetical order inside of its respective groups.
  - If the import line exceeds the 80 chars, then split it using parentheses to continue the import on the next line (aligning the imported items with the opening parenthesis).

## **Properties**

• Never use lambda functions:

```
# Good.
@property
def stores(self):
   return self.child.stores

# Bad.
stores = property(lambda self: self.child.stores)
```

• Try to use @property instead of get\_\* or is\_\* methods that don't require passing any parameter:

```
# Good.
@property
def terminology(self):
...
```

(continues on next page)

(continued from previous page)

```
@property
def is_monolingual(self):
    ...

# Also good.
def get_stores_for_language(self, language):
    ...

# Bad.
def get_terminology(self):
    ...

def is_monolingual(self):
    ...
```

• Always use @property instead of property (...), even for properties that also have a setter or a deleter:

```
# Good.
@property
def units(self):
 . . .
# Also good.
@property
def x(self):
  """I'm the 'x' property."""
  return self._x
@x.setter
def x(self, value): # Note: Method must be named 'x' too.
 self._x = value
@x.deleter
def x(self): # Note: Method must be named 'x' too.
 del self._x
# Bad.
def _get_units(self):
units = property(_get_units)
# Also bad.
def getx(self):
return self._x
def setx(self, value):
self.\_x = value
def delx(self):
 del self._x
x = property(getx, setx, delx, "I'm the 'x' property.")
```

## **Expressions and Statements**

## General whitespace rules

- No whitespace for unary operators that are not words (e.g.: -, ~ etc.) as well on the inner side of parentheses.
- Whitespace is placed between binary operators.

```
# Good.
exp = -1.05
value = (item_value / item_count) * offset / exp
value = my_list[index]
value = my_dict['key']

# Bad.
exp = -1.05
value = (item_value / item_count) * offset / exp
value = (item_value/item_count) * offset/exp
value = (item_value/item_count) * offset/exp
value = my_list[ index ]
value = my_dict ['key']
```

#### Slice notation

While PEP 8 calls for spaces around operators a = b + c this results in flags when you use a[b+1:c-1] but would allow the rather unreadable a[b+1:c-1] to pass. PEP 8 is rather quiet on slice notation.

- Don't use spaces with simple variables or numbers
- Use brackets for expressions with spaces between binary operators

```
# Good.
a[1:2]
a[start:end]
a[(start - 1):(end + var + 2)] # Brackets help group things and don't hide the slice
a[-1:(end + 1)]

# Bad.
a[start: end] # No spaces around:
a[start-1:end+var+2] # Insanely hard to read, especially when your expressions are more complex
a[start - 1:end + 2] # You lose sight of the fact that it is a slice
a[-1:end] # -1 is unary, no space
```

**Note:** String slice formatting is still under discussion.

## **Comparisons**

- Against arbitrary types: == and !=
- Against singletons with is and is not (e.g.: foo is not None)

• Never compare something with *True* or *False* (for example never do foo == False, do not foo instead)

# **Negated containment checks**

• Use foo not in bar instead of not foo in bar

## Instance checks

• isinstance(a, C) instead of type(A) is C, but try to avoid instance checks in general. Check for features.

## If statements

- Use () brackets around complex if statements to allow easy wrapping, don't use backslash to wrap an if statement.
- Wrap between and, or, etc.
- Keep not with the expression
- Use () alignment between expressions
- Use extra () to eliminate ambiguity, don't rely on an understanding of Python operator precedence rules.

```
# Good.
if length >= (upper + 2):
    ...

if (length >= 25 and
    string != "Something" and
    not careful):
    do_something()

# Bad.
if length >= upper + 2:
    ...

if (length...
    and string !=...
```

## **Naming Conventions**

Note: This has not been implemented or discussed. The Translate code is not at all consistent with these conventions.

- Class names: CamelCase, with acronyms kept uppercase (HTTPWriter and not HttpWriter)
- Variable names: lowercase\_with\_underscores
- Method and function names: lowercase\_with\_underscores
- Constants: UPPERCASE WITH UNDERSCORES
- precompiled regular expressions: name\_re

Protected members are prefixed with a single underscore. Double underscores are reserved for mixin classes.

To prevent name clashes with keywords, one trailing underscore may be appended. Clashes with builtins are allowed and **must not** be resolved by appending an underline to the name. If your code needs to access a shadowed builtin, rebind the builtin to a different name instead. Consider using a different name to avoid having to deal with either type of name clash, but don't complicate names with prefixes or suffixes.

# **Function and method arguments**

- Class methods: cls as first parameter
- Instance methods: self as first parameter

# 2.2 Documentation

We use Sphinx to generate our API and user documentation. Read the reStructuredText primer and Sphinx documentation as needed.

# 2.2.1 Special roles

We introduce a number of special roles for documentation:

```
• :issue: - links to a toolkit issue Github.
```

```
-:issue:`234` gives: issue 234-:issue:`broken <234>` gives: broken
```

• : opt: – mark command options and command values.

```
- : opt: `-P` gives -P- : opt: `--progress=dots` gives --progress=dots- : opt: `dots` gives dots
```

- :man: link to a Linux man page.
  - :man: `msgfmt` gives msgfmt

# 2.2.2 Code and command line highlighting

All code examples and format snippets should be highlighted to make them easier to read. By default Sphinx uses Python highlighting of code snippets (but it doesn't always work). You will want to change that in these situations:

• The examples are not Python e.g. talking about INI file parsing. In which case set the file level highlighting using:

```
.. highlight:: ini
```

• There are multiple different code examples in the document, then use:

```
.. code-block:: ruby
```

before each code block.

• Python code highlighting isn't working, then force Python highlighting using:

```
.. code-block:: python
```

**Note:** Generally we prefer explicit markup as this makes it easier for those following you to know what you intended. So use . . code-block:: python even though in some cases this is not required.

With command line examples, to improve readability use:

```
.. code-block:: console
```

Add \$ command prompt markers and # comments as required, as shown in this example:

```
$ cd docs
$ make html # Build all Sphinx documentation
$ make linkcheck # Report broken links
```

## 2.2.3 User documentation

This is documentation found in docs/ and that is published on Read the Docs. The target is the end user so our primary objective is to make accessible, readable and beautiful documents for them.

# 2.2.4 Docstrings

**Docstring conventions:** All docstrings are formatted with reStructuredText as understood by Sphinx. Depending on the number of lines in the docstring, they are laid out differently. If it's just one line, the closing triple quote is on the same line as the opening, otherwise the text is on the same line as the opening quote and the triple quote that closes the string on its own line:

```
def foo():
    """This is a simple docstring."""

def bar():
    """This is a longer docstring with so much information in there
    that it spans three lines. In this case the closing triple quote
    is on its own line.
    """
```

Please read PEP 257 (Docstring Conventions) for a general overview, the important parts though are:

- A docstring should have a brief one-line summary, ending with a period. Use Do this, Return that rather than Does ..., Returns ....
- If there are more details there should be a blank line between the one-line summary and the rest of the text. Use paragraphs and formatting as needed.
- Use reST field lists to describe the input parameters and/or return types as the last part of the docstring.
- Use proper capitalisation and punctuation.
- Don't restate things that would appear in parameter descriptions.

```
def addunit(self, unit):
    """Append the given unit to the object's list of units.
    (continues on next page)
```

2.2. Documentation 171

(continued from previous page)

```
This method should always be used rather than trying to modify the list manually.

:param Unit unit: Any object that inherits from :class:`Unit`.

"""
self.units.append(unit)
```

Parameter documentation: Document parameters using reST field lists as follows:

```
def foo(bar):
    """Simple docstring.

:param SomeType bar: Something
:return: Returns something
:rtype: Return type
    """
```

**Cross referencing code:** When talking about other objects, methods, functions and variables it is good practice to cross-reference them with Sphinx's Python cross-referencing.

Other directives: Use paragraph-level markup when needed.

**Note:** We still need to gather the useful ones that we want you to use and how to use them. E.g. how to talk about a parameter in the docstring. How to reference classes in the module. How to reference other modules, etc.

**Module header:** The module header consists of a utf-8 encoding declaration, copyright attribution, license block and a standard docstring:

```
# -*- coding: utf-8 -*-
#
... LICENSE BLOCK...
"""A brief description"""
```

**Deprecation:** Document the deprecation and version when deprecating features:

```
from translate.misc.deprecation import deprecated

@deprecated("Use util.run_fast() instead.")
def run_slow():
    """Run fast
    .. deprecated:: 1.5
        Use :func:`run_fast` instead.
    """
    run_fast()
```

## 2.2.5 Comments

## General:

• The # symbol (pound or hash) is used to start comments.

- A space must follow the # between any written text.
- Line length must be observed.
- Inline comments are preceded by two spaces.
- Write sentences correctly: proper capitalisation and punctuation.

```
# Good comment with space before and full sentence.
statement # Good comment with two spaces

#Bad comment no space before
statement # Bad comment, needs two spaces
```

**Docstring comments:** Rules for comments are similar to docstrings. Both are formatted with reStructuredText. If a comment is used to document an attribute, put a colon after the opening pound sign (#):

```
class User(object):
    #: the name of the user as unicode string
    name = Column(String)
    #: the shal hash of the password + inline salt
    pw_hash = Column(String)
```

# 2.3 Building

## 2.3.1 UNIX

# 2.3.2 Windows

# Requirements

- Innosetup
- py2exe

Consult the README in the source distribution for the build dependencies.

## **Building Python packages with C extensions under Windows**

In order to build modules which have C extensions, you will need either the Visual Studio C++ compiler or MinGW.

Make sure that your Visual Studio C++ or MinGW program path is part of your system's program path, since the Python build system requires this.

To build and install a package with MinGW, you need to execute:

```
python setup.py build -c mingw32 install
```

from the command line.

To build a Windows installer when using MinGW, execute:

```
python setup.py build -c mingw32 bdist_wininst
```

2.3. Building 173

## **Building**

Simply execute:

```
python setup.py innosetup
```

The generated file can be found under translate-toolkit-<version>\Output (where <version> is the software version).

# 2.4 Testing

Our aim is that all new functionality is adequately tested. Adding tests for existing functionality is highly recommended before any major reimplementation (refactoring, etcetera).

We use py.test for (unit) testing. You need at least pytest  $\geq$  2.2.

To run tests in the current directory and its subdirectories:

```
$ py.test # runs all tests
$ py.test storage/test_dtd.py # runs just a single test module
```

We use several py.test features to simplify testing, and to suppress errors in circumstances where the tests cannot possibly succeed (limitations of tests and missing dependencies).

# 2.4.1 Skipping tests

Pytest allows tests, test classes, and modules to be skipped or marked as "expected to fail" (xfail). Generally you should *skip* only if the test cannot run at all (throws uncaught exception); otherwise *xfail* is preferred as it provides more test coverage.

# importorskip

Use the builtin \_pytest.runner.importorskip function to skip a test module if a dependency cannot be imported:

```
from pytest import importorskip
importorskip("vobject")
```

If *vobject* can be imported, it will be; otherwise it raises an exception that causes pytest to skip the entire module rather than failing.

# skipif

Use the skipif decorator to mark tests to be skipped unless certain criteria are met. The following skips a test if the version of *mymodule* is too old:

```
import mymodule
@pytest.mark.skipif("mymodule.__version__ < '1.2'")
def test_function():
    ...</pre>
```

You can apply this decorator to classes as well as functions and methods.

It is also possible to skip an entire test module by creating a pytestmark static variable in the module:

```
# mark entire module as skipped for py.test if no indexer available
pytestmark = pytest.mark.skipif("noindexer")
```

#### xfail

Use the xfail decorator to mark tests as expected to fail. This allows you to do the following:

- · Build tests for functionality that we haven't implemented yet
- · Mark tests that will fail on certain platforms or Python versions
- Mark tests that we should fix but haven't got round to fixing yet

The simplest form is the following:

```
from pytest import pytest.mark

@mark.xfail
def test_function():
    ...
```

You can also pass parameters to the decorator to mark expected failure only under some condition (like *skipif*), to document the reason failure is expected, or to actually skip the test:

```
@mark.xfail("sys.version_info >= (3,0)") # only expect failure for Python 3
@mark.xfail(..., reason="Not implemented") # provide a reason for the xfail
@mark.xfail(..., run=False) # skip the test but still regard it as xfailed
```

# 2.4.2 Testing for Warnings

## deprecated call

The builtin deprecated\_call() function checks that a function that we run raises a DeprecationWarning:

```
from pytest import deprecated_call

def test_something():
    deprecated_call(function_to_run, arguments_for_function)
```

## recwarn

The *recwarn plugin* allows us to test for other warnings. Note that recwarn is a funcargs plugin, which means that you need it in your test function parameters:

```
def test_example(recwarn):
    # do something
    w = recwarn.pop()
    # w.{message, category, filename, lineno}
    assert 'something' in str(w.message)
```

2.4. Testing 175

# 2.5 Command Line Functional Testing

Functional tests allow us to validate the operation of the tools on the command line. The execution by a user is simulated using reference data files and the results are captured for comparison.

The tests are simple to craft and use some naming magic to make it easy to refer to test files, stdout and stderr.

# 2.5.1 File name magic

We use a special naming convention to make writing tests quick and easy. Thus in the case of testing the following command:

```
$ moz2po -t template.dtd translations.po translated.dtd
```

Our test would be written like this:

```
$ moz2po -t $one $two $out
```

Where Sone and Stwo are the input files and Sout is the result file that the test framework will validate.

The files would be called:

File	Function	Variable	File naming conven-
			tions
test_moz2po_help.sh	Test script	•	test_\${command}_\${description}.sh
test_moz2po_help/one.dtd	Input	\$one	\${testname}/\${variable}.\${extension}
test_moz2po_help/two.po	Input	\$two	\${testname}/\${variable}.\${extension}
test_moz2po_help/out.dtd	Output	\$out	\${testname}/\${variable}.\${extension}
test_moz2po_help/stdout.tx	t Output	\$stdout	\${testname}/\${variable}.\${extension}
test_moz2po_help/stderr.txt	Output	\$stderr	\${testname}/\${variable}.\${extension}

**Note:** A test filename must start with test\_ and end in .sh. The rest of the name may only use ASCII alphanumeric characters and underscore \_.

The test file is placed in the tests/ directory while data files are placed in the tests/data/\${testname} directory.

There are three standard output files:

- 1. \$out the output from the command
- 2. \$stdout any output given to the user
- 3. \$stderr any error output

The output files are available for checking at the end of the test execution and a test will fail if there are differences between the reference output and that achieved in the test run.

You do not need to define reference output for all three, if one is missing then checks will be against /dev/null.

There can be any number of input files. They need to be named using only ASCII characters without any punctuation. While you can give them any name we recommend using numbered positions such as one, two, three. These are converted into variables in the test framework so ensure that none of your choices clash with existing bash commands and variables.

Your test script can access variables for all of your files so e.g. moz2po\_conversion/one.dtd will be referenced as \$one and output moz2po\_conversion/out.dtd as \$out.

## 2.5.2 Writing

The tests are normal bash scripts so they can be executed on their own. A template for a test is as follows:

```
#!/bin/bash
# Import the test framework
source $(basename $0)/test.inc.sh
# You can put any extra preperation here
# Your actual command line to test No need for redirecting to /dev/stdout as
# the test framework will do that automatically
myprogram $one $two -o $out
# Check that the results of the test match your reference resulst
check_results # does start_check and diff_all
# OR do the following
# start_checks - begin checking
# has_stdout|has_stderr|has $file - checks that the file exists we don't care for ...
# startswith $file|startswith_stderr|startswith_stdout - the output starts with some.
# startswithi $file|startswithi_stderr|startswithi_stdout - case insensitive_
\hookrightarrow startswith
# end_checks
```

For simple tests, where we diff output and do the correct checking of output files, simply use <code>check\_results</code>. More complex tests need to wrap tests in <code>start\_checks</code> and <code>end\_checks</code>.

```
start_checks
has $out
containsi_stdout "Parsed:"
end_checks
```

You can make use of the following commands in the start\_checks scenario:

Command	Description
has \$file	\$file was output and it not empty
has_stdout	stdout is not empty
has_stderr	stderr is not empty
startswith \$file "String"	\$file starts with "String"
startswithi \$file "String"	\$file starts with "String" ignoring case
startswith_stdout "String"	stdout starts with "String"
startswithi_stdout "String"	stdout starts with "String" ignoring case
startswith_stderr "String"	stderr starts with "String"
startswithi_stderr "String"	stderr starts with "String" ignoring case
contains \$file "String"	\$file contains "String"
containsi \$file "String"	\$file contains "String" ignoring case
contains_stdout "String"	stdout contains "String"
containsi_stdout "String"	stdout contains "String" ignoring case
contains_stderr "String"	stderr contains "String"
containsi_stderr "String"	stderr contains "String" ignoring case
endswith \$file "String"	\$file ends with "String"
endswithi \$file "String"	\$file ends with "String" ignoring case
endswith_stdout "String"	stdout ends with "String"
endswithi_stdout "String"	stdout ends with "String" ignoring case
endswith_stderr "String"	stderr ends with "String"
endswithi_stderr "String"	stderr ends with "String" ignoring case

#### -prep

If you use the -prep options on any test then the test will change behavior. It won't validate the results against your reference data but will instead create your reference data. This makes it easy to generate your expected result files when you are setting up your test.

# 2.6 Contributing

We could use your help. If you are interesting in contributing then please join us on our Gitter development channel.

Here are some idea of how you can contribute

- Test help us test new candidate releases before they are released
- *Debug* check bug reports, create tests to highlight problems
- Develop add your Python developer skills to the mix
- Document help make our docs readable, useful and complete

Below we give you more detail on these:

## 2.6.1 Testing

Before we release new versions of the Toolkit we need people to check that they still work correctly. If you are a frequent user you might want to start using the release candidate on your current work and report any errors before we release them.

Compile and install the software to see if we have any platform issues:

./setup.py install

Check for any files that are missing, tools that were not installed, etc.

Run unit tests to see if there are any issues. Please report any failures.

Finally, simply work with the software. Checking all your current usage patterns and report problems.

## 2.6.2 Debugging

- Make sure your familiar with the bug reporting guidelines.
- Create a login for yourself at https://github.com
- · Then choose an issue

Now you need to try and validate the bug. Your aim is to confirm that the bug is either fixed, is invalid or still exists.

If its fixed please close the bug and give details of how when it was fixed or what version you used to validate it as corrected.

If you find that the bug reporter has made the incorrect assumptions or their suggestion cannot work. Then mark the bug as invalid and give reasons why.

The last case, an existing bug is the most interesting. Check through the bug and do the following:

- Fix up the summary to make it clear what the bug is
- · Create new bugs for separate issues
- Set severity level and classifications correctly
- Add examples to reproduce the bug, or make the supplied files simpler
- If you can identify the bug but not fix it then explain what needs fixing
- · Move on to the next bug

## 2.6.3 Developing

Don't ignore this area if you feel like you are not a hotshot coder!

You will need some Python skills, this is a great way to learn.

Here are some ideas to get you going:

- Write a test to expose some bug
- Try to fix the actual code to fix your bug
- · Add a small piece of functionality that helps you
- Document the methods in especially the base class and derived classes
- Add a *format* type and converters
- Add more features to help our formats conform to the standards

You will definitely need to be on the Development channel

Now is the time to familiarise yourself with the *developers guide*.

2.6. Contributing 179

## 2.6.4 Documenting

This is the easy one. Login to the wiki and start!

The key areas that need to be looked at are:

- Do the guides to each tool cover all command line options
- Are the examples clear for the general cases
- Is the tools use clear
- · In the Use cases, can we add more, do they need updating. Has upstream changed its approach

After that and always:

- Grammar
- Spelling
- Layout

# 2.7 Translate Toolkit Developers Guide

The goal of the translate toolkit is to simplify and unify the process of translation.

## 2.7.1 History

The initial toolkit was designed to convert Mozilla .dtd and .properties files into Gettext PO format. The logic was not that PO was in any way superior but that by simplifying the translations process i.e. allowing a translator to use one format and one tool that we could get more people involved and more translators.

The tools have now evolved to include other formats such as OpenOffice.org and the goal is still to migrate various formats to a common format, PO and in the future XLIFF as more tools become available for that format.

These tools we group as converters. Along the way we developed other tools that allowed us to manipulate PO files and check them for consistency. As we evolved the converter tools we have also improved and abstracted the classes that read the various file types. In the future we hope to define these better so that we have a more or less stable API for converters.

## 2.7.2 Resources

#### Git access

Translate Toolkit uses Git as a Version Control System. You can directly clone the translate repository or fork it at GitHub.

git clone https://github.com/translate/translate.git

#### **Issues**

https://github.com/translate/translate/issues

#### Communication

- Development no support related questions
- Help

## 2.7.3 Working with Bugzilla

When you close bugs ensure that you give a description and git hash for the fix. This ensures that the reporter or code reviewer can see your work and has an easy method of finding your fix. This is made easier by GitHub's Bugzilla integration.

## **Automated Bugzilla update from commits**

Github will post comments on Bugzilla bugs when the commit messages make references to the bug by its bug number.

• Bugs are recognised by the following format (which are case-insensitive):

```
Bug 123
```

· Multiple bugs can be specified by separating them with a comma, ampersand, plus or "and":

```
Bug 123, 124 and 125
```

- Commits to all branches will be processed.
- If there is a "fix", "close", or "address" before the bug then that bug is closed.

```
Fix bug 123
```

## 2.7.4 Source code map

The source code for the tools is hosted on GitHub. This rough map will allow you to navigate the source code tree:

- convert convert between different formats and PO format
- filters *pofilter* and its helper functions (badly named, it is really a checking tool)
- storage all base file formats: XLIFF, .properties, OpenOffice.org, TMX, etc.
- misc various helper functions
- tools all PO manipulation programs: pocount, pogrep, etc
- lang modules with data / tweaks for various languages
- search translation memory, terminology matching, and indexing / searching
- share data files

## 2.7.5 Setup

The toolkit is installed by running:

```
./setup.py install
```

As root

The various setup options are yours to explore

## 2.7.6 General overview of the programs

Each tool in the toolkit has both a core program and a command line wrapper. For example the oo2po converter:

- oo2po the command line tool
- oo2po.py the core program

This is done so that the tools can be used from within the Pootle server thus reusing the toolkit easily.

#### **Command line options**

Getting lost with the command line options? Well you might want to volunteer to move some of them into configuration files. But in terms of programming you might be confused as to where they are located. Many of the command line options are implemented in each tool. Things such as --progress and --errorlevel are used in each program. Thus these are abstracted in **misc/optrecurse.py**. While each tools unique command line options are implemented in **xxx.py**.

#### 2.7.7 Converters

The converters each have a class that handles the conversion from one format to another. This class has one important method **convertfile** which handles the actual conversion.

A function **convertXXX** manages the conversion for the command line equivalent and essentially has at least 3 parameters: inputfile, outputfile and templatefile. It itself will call the conversion class to handle conversion of individual files. Recursing through multiple files is handled by the optrecurse.py logic.

The converters **main** function handles any unique command line options.

Where we are headed is to get to a level where the storage formats themselves are more aware of themselves and their abilities. Thus the converter could end up as one program that accepts storage format plugins to convert from anything to almost anything else. Although our target localisation formats are PO and XLIFF only.

If you want to create a new converter it is best to look at a simple instance such as csv2tbx or txt2po and their associated storage classes. The storage base class documentation will give you the information you need for the storage class implementation.

## **2.7.8 Tools**

The tools in some way copy the logic of the converters. We have a class so that we can reuse a lot of the functionality in Pootle. We have a core function that take: input, output and templates. And we have a **main** function to handle the command line version.

*pocount* should be converted to this but does not follow this conventions. In fact pocount should move the counting to the storage formats to allow any format to return its own word count.

#### 2.7.9 Checks

There's really only one, *pofilter*. But there are lots of helper functions for pofilter. pofilters main task is to check for errors in PO or XLIFF files. Here are the helper file and their descriptions.

- autocorrect.py when using –-autocorrect it will attempt some basic corrections found in this file
- checks.py the heart. This contains: the actual checks and their error reports, and defined variables and accelerators for e.g, --mozilla
- decorations.py various helper functions to identify accelerators, variables and markers
- helpers.py functions used by the tests
- prefilters.py functions to e.g. remove variables and accelerators before applying tests to the PO message pofilter is now relatively mature. The best areas for contributions are:
  - · more tests
  - · language specific configuration files
  - tests for the tests so we don't break our good tests
  - defining a config files scheme to do cool stuff off of the command line. Globally enable or disable tests based on language, etc
  - some approach to retesting that would remove '# (pofilter)' failure markings if the test now passes.
  - ability to mark false positives

The *API documentation* is a good start if you want to add a new tests. To add a new language have a look at a language you understand amongst those already implemented.

## **2.7.10 Storage**

These are the heart of the converters. Each destination storage format is implemented in its own file. Up until toolkit version 0.8, there was no formally defined API (the tools have been evolving and only recently stabilised), but they generally followed this structure. These classes are defined:

- XXelement handles the low level individual elements of the file format. e.g. PO message, CSV records, DTD elements
- XXfile handles the document or file level of the format. Eg a PO file, a CSV file a DTD file
  - fromlines read in a file and initialise the various elements
  - tolines convert the elements stored in XXelements and portions in XXfile to a raw file in that format

In the XML based formats e.g. TMX, XLIFF and HTML there is usually just an extended parser to manage the file creation.

Within each storage format there are many helper functions for escaping and managing the unique features of the actual format.

You can help by:

- abstracting more of the functions and documenting that so that we can get a better API
- adding other formats and converters e.g. .DOC, .ODF and others
- helping us move to a position where any format should convert to the base format: PO and in the future XLIFF without having to create a specific converter wrapper.
- Ensuring that our formats conform to the standards

#### **Base Classes**

From toolkit 0.9 onwards, we are moving towards basing all storage formats on a set of *base classes*, in the move to a universal API. We're also fixing things so that escaping is much more sane and handled within the class itself not by the converters.

In base classes we have different terminology

- XXXunit = XXXelement
- XXXstore = XXXfile

We have also tried to unify terminology but this has been filtered into the old classes as far as possible.

# 2.8 Making a Translate Toolkit Release

This page is divided in three sections. The first one lists the tasks that must be performed to get a valid package. The second section includes a list of tasks to get the package published and the release announced. The third one lists and suggests some possible cleanup tasks to be done after releasing.

Note: Please note that this is not a complete list of tasks. Please feel free to improve it.

## 2.8.1 Create the package

The first steps are to create and validate a package for the next release.

#### Get a clean checkout

We work from a clean checkout to ensure that everything you are adding to the build is what is in the repository and doesn't contain any of your uncommitted changes. It also ensures that someone else could replicate your process.

```
$ git clone git@github.com:translate/translate.git translate-release
$ cd translate-release
$ git submodule update --init
```

#### **Check copyright dates**

Update any copyright dates in docs/conf.py:copyright and anywhere else that needs fixing.

```
$ git grep 2013 # Should pick up anything that should be examined
```

#### Create release notes

We create our release notes in reStructured Text, since we use that elsewhere and since it can be rendered well in some of our key sites.

First we need to create a log of changes in the Translate Toolkit, which is done generically like this:

```
$ git log $previous_version..HEAD > docs/releases/$version.rst
```

Or a more specific example:

```
$ git log 1.10.0..HEAD > docs/releases/1.11.0-rc1.rst
```

Edit this file. You can use the commits as a guide to build up the release notes. You should remove all log messages before the release.

**Note:** Since the release notes will be used in places that allow linking we use links within the notes. These should link back to products websites (Virtaal, Pootle, etc.), references to Translate and possibly bug numbers, etc.

Read for grammar and spelling errors.

**Note:** When writing the notes please remember:

- 1. The voice is active. 'Translate has released a new version of the Translate Toolkit', not 'A new version of the Translate Toolkit was released by Translate'.
- 2. The connection to the users is human not distant.
- 3. We speak in familiar terms e.g. "I know you've been waiting for this release" instead of formal.

We create a list of contributors using this command:

```
$ git log 1.10.0..HEAD --format='%aN, ' | awk '{arr[$0]++} END{for (i in arr){print_

→arr[i], i;}}' | sort -rn | cut -d\ -f2-
```

#### Up version numbers

Update the version number in:

- translate/\_\_version\_\_.py
- docs/conf.py

In translate/\_\_version\_\_.py, bump the build number if anybody used the Translate Toolkit with the previous number, and there have been any changes to code touching stats or quality checks. An increased build number will force a Translate Toolkit user, like Pootle, to regenerate the stats and checks.

For docs/conf.py change version and release.

**Todo:** FIXME - We might want to consolidate the version and release info so that we can update it in one place.

The version string should follow the pattern:

```
$MAJOR-$MINOR-$MICRO[-$EXTRA]
```

E.g.

```
1.10.0
0.9.1-rc1
```

\$EXTRA is optional but all the three others are required. The first release of a \$MINOR version will always have a \$MICRO of .0. So 1.10.0 and never just 1.10.

**Note:** You probably will have to adjust the output of some of the functional tests, specifically the manpage ones, to use the right new version.

#### **Build the package**

Building is the first step to testing that things work. From your clean checkout run:

```
$ mkvirtualenv build-ttk-release
(build-ttk-release)$ pip install --upgrade setuptools pip
(build-ttk-release)$ pip install -r requirements/dev.txt
(build-ttk-release)$ make build
(build-ttk-release)$ deactivate
```

This will create a tarball in dist/ which you can use for further testing.

**Note:** We use a clean checkout just to make sure that no inadvertent changes make it into the release.

#### Test install and other tests

The easiest way to test is in a virtualenv. You can test the installation of the new release using:

```
$ mkvirtualenv test-ttk-release
(test-ttk-release)$ pip install --upgrade setuptools pip
(test-ttk-release)$ pip install dist/translate-toolkit-$version.tar.bz2
```

You can then proceed with other tests such as checking:

- 1. Documentation is available in the package
- 2. Converters and scripts are installed and run correctly:

```
(test-ttk-release)$ moz2po --help
(test-ttk-release)$ php2po --version
(test-ttk-release)$ deactivate
$ rmvirtualenv test-ttk-release
```

3. Meta information about the package is correct. This is stored in setup.py, to see some options to display meta-data use:

```
$ ./setup.py --help
```

Now you can try some options like:

```
$ ./setup.py --name
$ ./setup.py --version
$ ./setup.py --author
$ ./setup.py --author-email
$ ./setup.py --url
$ ./setup.py --license
$ ./setup.py --description
$ ./setup.py --long-description
$ ./setup.py --classifiers
```

The actual descriptions are taken from translate/\_\_init\_\_.py.

#### 2.8.2 Publish the new release

Once we have a valid package it is necessary to publish it and announce the release.

#### Tag and branch the release

You should only tag once you are happy with your release as there are some things that we can't undo. You can safely branch for a stable/ branch before you tag.

```
$ git checkout -b stable/2.2.x
$ git push origin stable/2.2.x
$ git tag -a 2.2.5 -m "Tag version 2.2.5"
$ git push --tags
```

#### Release documentation

We need a tagged release before we can do this. The docs are published on Read The Docs.

https://readthedocs.org/dashboard/translate-toolkit/versions/

Use the admin pages to flag a version that should be published.

#### **Publish on PyPI**

**Note:** You need a username and password on Python Package Index (PyPI) and have rights to the project before you can proceed with this step.

These can be stored in \$HOME/.pypirc and will contain your username and password. Check Create a PyPI account for more details.

Run the following to publish the package on PyPI:

```
$ workon build-ttk-release
(build-ttk-release)$ pip install --upgrade pyopenssl ndg-httpsclient pyasn1 twine
(build-ttk-release)$ twine upload dist/translate-toolkit-*
(build-ttk-release)$ deactivate
$ rmvirtualenv build-ttk-release
```

#### Create a release on Github

• https://github.com/translate/translate/releases/new

You will need:

- Tarball of the release
- · Release notes in Markdown

Do the following to create the release:

1. Draft a new release with the corresponding tag version

- 2. Convert the major changes (no more than five) in the release notes to Markdown with Pandoc. Bugfix releases can replace the major changes with *This is a bugfix release for the X.X.X branch*.
- 3. Add the converted major changes to the release description
- 4. Include at the bottom of the release description a link to the full release notes at Read The Docs
- 5. Attach the tarball to the release
- 6. Mark it as pre-release if it's a release candidate

#### **Update Translate Toolkit website**

We use github pages for the website. First we need to checkout the pages:

```
$ git checkout gh-pages
```

- 1. In \_posts/ add a new release posting. This is in Markdown format (for now), so we need to change the release notes .rst to .md, which mostly means changing URL links from `xxx <link>`\_ to [xxx] (link).
- 2. Change \$version as needed. See download.html, \_config.yml and egrep -r \$old\_release
- 3. **git commit** and **git push** changes are quite quick, so easy to review.

#### Announce to the world

Let people know that there is a new version:

- 1. Announce on mailing lists **using plain text** emails using the same text (adjusting what needs to be adjusted) used for the *Create a release on Github* description:
  - · translate-announce@lists.sourceforge.net
  - translate-pootle@lists.sourceforge.net
  - translate-devel@lists.sourceforge.net
- 2. Adjust the Pootle gitter channel topic. Use /topic [new topic] to change the topic.
- 3. Adjust the #pootle channel notice. Use /topic [new topic] to change the topic. It is easier if you copy the previous topic and adjust it.

**Note:** You might need to identify yourself by using /msg nickserv identify [password] so the IRC server knows you in order to check if you have enough permissions.

- 4. Email important users
- 5. Tweet about it
- 6. Update Translate Toolkit's Wikipedia page

## 2.8.3 Post-Releasing Tasks

These are tasks not directly related to the releasing, but that are nevertheless completely necessary.

#### Bump version to N+1-alpha1

If this new release is a stable one, bump the version in master to  $\{N+1\}$ -alpha1. The places to be changed are the same ones listed in *Up version numbers*. This prevents anyone using master being confused with a stable release and we can easily check if they are using master or stable.

**Note:** You probably will have to adjust the output of some of the functional tests, specifically the manpage ones, to use the right new version.

#### Add release notes for dev

After updating the release notes for the about to be released version, it is necessary to add new release notes for the next release, tagged as dev.

## Other possible steps

Some possible cleanup tasks:

- Remove your translate-release checkout.
- Update and fix these releasing notes:
  - Make sure these releasing notes are updated on master.
  - Discuss any changes that should be made or new things that could be added.
  - Add automation if you can.

We also need to check and document these if needed:

- Change URLs to point to the correct docs: do we want to change URLs to point to the \$version docs rather then latest?
- Building on Windows, building for other Linux distros.
- · Communicating to upstream packagers.

# 2.9 Deprecation of Features

From time to time we need to deprecate functionality, this is a guide as to how we implement deprecation.

# 2.9.1 Types of deprecation

- 1. Misspelled function
- 2. Renamed function
- 3. Deprecated feature

#### 2.9.2 Period of maintenance

Toolkit retains deprecated features for a period of two releases. Thus features deprecated in 1.7.0 are removed in 1.9.0.

#### 2.9.3 Documentation

Use the @deprecated decorator with a comment and change the docstring to use the Sphinx deprecation syntax.

```
@deprecated("Use util.run_fast() instead.")
def run_slow():
    """Run slowly
    .. deprecated:: 1.9.0
        Use :func:`run_fast` instead.
    """
    run_fast() # Call new function if possible
```

## 2.9.4 Implementation

Deprecated features should call the new functionality if possible. This may not always be possible, such as the cases of drastic changes. But it is the preferred approach to reduce maintenance of the old code.

#### 2.9.5 Announcements

**Note:** This applies only to feature deprecation and renamed functions. Announcements for corrections are at the coders discretion.

- 1. On **first release with deprecation** highlight that the feature is deprecated in this release and explain reasons and alternate approaches.
- 2. On **second relase** warn that the feature will be removed in the next release.
- 3. On **third release** remove the feature and announce removal in the release announcements.

Thus by examples:

**Translate Toolkit 1.9.0:** The run\_slow function has been deprecated and replaced by the faster and more correct run\_fast. Users of run\_slow are advised to migrate their code.

**Translate Toolkit 1.10.0:** The run\_slow function has been deprecated and replaced by run\_fast and will be removed in the next version. Users of run\_slow are advised to migrate their code.

Translate Toolkit 1.11.0: The run\_slow function has been removed, use run\_fast instead.

# CHAPTER 3

# **Additional Notes**

Changelog and legal information are included here.

## 3.1 Release Notes

The following are release notes for the Translate Toolkit releases.

These are the changes that have happened in the Translate Toolkit and which may impact you. If you use Pootle, Virtaal or any other application that makes use of the Translate Toolkit you may want to familiarize yourself with these changes.

## 3.1.1 Final releases

#### **Translate Toolkit 2.4.0**

Released on 24 June 2019

This release contains improvements and bug fixes.

#### **Changes**

#### **Formats and Converters**

- PO
  - Allow unicode in PO headers (issue 3896).
  - Improve finding out newline format for a file.
  - Strip UTF-8 BOM from input (issue 1640).
- XLIFF

- Adjustments on how output is indented (issue 3424).
- Properties
  - Do not fail when parsing empty file.
- · Android resources
  - Multiple adjustments so output is closer to Android Studio's.
- YAML
  - Switched to *ruamel.yaml* to simplify codebase and support YAML 1.2.
  - Added support for Ruby plurals.
  - Fixed handling dict in list (issue 3895).
  - Fixed parsing of empty YAML file.
- JSON
  - Fixed serialization of JSON arrays.
  - Placeholders are now kept in WebExtension dialect round trip conversion.
- RESX
  - Several improvements on formatting to align with Visual Studio's output.
- TS
  - Improved tags indentation.
  - Added support for new vanished type.
- Flat XML
  - Added support for this new format including *flatxml2po* and *po2flatxml* converters (issue 3776).
- CSV
  - No longer hardcode escape character (issue 3246).
  - Rewrote default dialect to make it more flexible.
- web2py
  - Updated converters code.
- Subtitles
  - Initialize duration on subtitle unit \_\_init\_\_.

#### **Tools**

• Tmserver: Fixed execution of unit API on Python 3.

#### Languages

- Updated plural definitions to CLDR 35.0.
- Removed trailing semicolon in Romanian plural definition.

#### **Placeables**

• Allow any character for Python mapping keys in *PythonFormattingPlaceable*.

## **API** changes

- Altered storage code to have a consistent API for removenotes.
- Removed dependency on diff-match-patch.
- Removed embedded CherryPy wsgi server.
- Removed deprecated *has\_key* implementation.

#### General

- Dropped no longer supported Python 3.3 and Python 3.4.
- Minor docs improvements.
- Updated requirements.
- Added and updated tests.

#### **Contributors**

This release was made possible by the following people:

Michal Čihař, Leandro Regueiro, Vinyl Darkscratch, Vitaly Novichkov, Stuart Prescott, Alex Tomkins, Darío Hereñú, BhaaL.

And to all our bug finders and testers, a Very BIG Thank You.

## **Translate Toolkit 2.3.1**

Released on 06 October 2018

This release contains improvements and bug fixes.

#### **Changes**

#### **Formats and Converters**

- PO
  - Fixed parsing of files with mixed newlines n and r.
- · Properties
  - Fixed escaping of serialized string for Joomla dialect.
  - Fixed loading of OS X strings dialect files having multiline strings.
- · Android resources
  - Correctly handle escaping of question mark.

- PHP
  - Improved handling of unit name.
  - Handle [] style arrays.
  - Added support for *return* statement.
- YAML
  - Consistent handling of *int* and *bool*.
  - Fixed serialization of empty keys.
- JSON
  - Nested values ordering is now preserved.
- TMX
  - Avoid mentioning *po2tmx* in creation tool.
- RESX
  - New unit elements are now properly indented.
- INI
  - Enabled support for Python3 provided that patched *iniparse* library is available.
- RC
  - Altered to remove r before parsing.

#### **API changes**

- Use backports.csv module on Python 2 to align the behavior with Python 3 and drop many hacks.
- Removed deprecated *getoutput* methods deprecated in version 2.0.0.
- Added new deprecations:
  - Deprecated setsource, getsource, gettarget and settarget methods in favor of source and target properties
    for all storage classes, except LISAunit and its subclasses since for those these methods do actually accept
    additional arguments so can't just be always replaced by some property.
  - xliffunit: Deprecated get\_rich\_source in favor of rich\_source property
- Defined for all unit classes the *rich\_source* and *rich\_target* properties without using methods. Since the old methods were private they were directly removed without deprecating them.

#### General

- Refactored more converters to increase readability and use a common pattern which will allow to further refactor repeated code
- Code cleanups and simplification
- Updated requirements
- Tests:
  - Added plenty of tests
  - Tests cleanups and fixes

#### **Contributors**

This release was made possible by the following people:

Leandro Regueiro, Michal Čihař, BhaaL, Mark Jansen, Stuart Prescott, David Hontecillas.

And to all our bug finders and testers, a Very BIG Thank You.

#### **Translate Toolkit 2.3.0**

Released on 18 March 2018

This release contains improvements and bug fixes.

#### **Changes**

#### **Formats and Converters**

- PO
  - Avoid escaping some characters (()/:,) that don't need to be escaped
  - Wrap lines on / like Gettext
  - Lines can be wrapped at specified length
  - MO units now allow to set the unit context
  - Always URL-encode locations
- PHP
  - Full rewrite of the PHP format using phply:
    - \* Fixes multiple issues
    - \* Brings support for new dialects: heredoc, nowdoc, short array syntax and nested arrays.
- YAML
  - Added yaml2po and po2yaml converters
  - Fixed some minor bugs
  - Documented unsupported features
- JSON
  - Refactored the storage classes to get rid of repeated shared code, reduce memory usage and get readable representation of the units.
- txt
  - Added --no-segmentation flag to txt2po

#### **Tools**

• Removed translate.convert.poreplace

#### General

- Improved support for Windows
- Refactored multiple converters to increase readability and use a common pattern which will allow to further refactor repeated code
- Tests:
  - Enabled testing on Windows
  - Added more tests
  - Plenty of tests cleanups and fixes
- Docs:
  - Updated docs on installation
  - Improved automatic generation of docs on factories

#### **Contributors**

This release was made possible by the following people:

Leandro Regueiro, Michal Čihař, Stuart Prescott, Nick Schonning, Johannes Marbach, andreistefan87, Alejandro Mantecon Guillen.

And to all our bug finders and testers, a Very BIG Thank You.

## **Translate Toolkit 2.2.5**

Released on 01 September 2017

This release contains improvements and bug fixes.

#### **Changes**

#### **Formats and Converters**

- XLIFF
  - Fixed bug when adding new units to XLIFF store.
- JSON
  - Added support for i18next JSON dialect.
  - Improved WebExtension JSON dialect support.

#### **Contributors**

This release was made possible by the following people:

Michal Čihař, Leandro Regueiro, Ryan Northey.

And to all our bug finders and testers, a Very BIG Thank You.

#### **Translate Toolkit 2.2.4**

Released on 31 July 2017

This release contains improvements and bug fixes.

#### **Changes**

#### **Formats and Converters**

- XLIFF
  - Added support for .xliff extension in all converters and tools that support .xlf extension.
- JSON
  - Added support for nested JSON.
  - Added support for WebExtension JSON dialect.
- txt
  - po2txt skips obsolete and non-translatable strings.

#### **Filters and Checks**

- The *puncspace* check now strips Bidi markers chars before processing.
- Added ReducedChecker checker to list of checkers.

## **API changes**

- Language and country default to *common\_name* if available.
- Added function to retrieve all language classes.

#### **Contributors**

This release was made possible by the following people:

Dwayne Bailey, Leandro Regueiro, Michal Čihař, Rimas Kudelis, Ludwig Nussel, Stuart Prescott.

And to all our bug finders and testers, a Very BIG Thank You.

#### **Translate Toolkit 2.2.3**

Released on 20 June 2017

This release contains improvements and bug fixes.

## **Changes**

• Added MinimalChecker and ReducedChecker checkers.

#### **Contributors**

This release was made possible by the following people:

Rimas Kudelis, Leandro Regueiro.

And to all our bug finders and testers, a Very BIG Thank You.

#### **Translate Toolkit 2.2.2**

Released on 20 June 2017

This release contains improvements and bug fixes.

## **Changes**

• Fixed resolving of country names translations.

#### **Contributors**

This release was made possible by the following people:

Ryan Northey, Leandro Regueiro.

And to all our bug finders and testers, a Very BIG Thank You.

#### **Translate Toolkit 2.2.1**

Released on 19 June 2017

This release contains many improvements and bug fixes.

#### **Major changes**

- Refactored functions for resolving language/country names translation to be memory efficient.
- Improvements for ts and subtitles formats.
- Added -preserveplaceholders argument to podebug.
- Fixed Montenegrin language name.

## **Detailed changes**

#### **Formats and Converters**

- ts
- Write quotes as entities
- Remove not necessary encoding/decoding to UTF-8
- Subtitles
  - Avoid errors when subtitle support is missing

#### **Tools**

• Added -preserveplaceholders argument to podebug to avoid rewriting placeholders

## Languages

• Fixed Montenegrin language name.

## **API changes**

· Refactored functions for resolving language/country names translation to be memory efficient

#### General

- Use gzip for packaging
- Python 3 fixes
- · Added more tests

... and loads of general code cleanups and of course many many bugfixes.

#### **Contributors**

This release was made possible by the following people:

Michal Čihař, Leandro Regueiro, Ryan Northey, Robbie Cole, Kai Pastor, Dwayne Bailey.

And to all our bug finders and testers, a Very BIG Thank You.

#### **Translate Toolkit 2.2.0**

Released on 15 June 2017

This release contains many improvements and bug fixes.

#### **Major changes**

- Avoid resolving external entities while parsing XML.
- Improvements for Android, ts and resx formats.
- Added support for PHP nested arrays.
- Added Kabyle language

## **Detailed changes**

## Requirements

- Updated requirements.
- Added *pycountry* recommended requirement for localized language names.

#### **Formats and Converters**

- XML formats
  - Avoid resolving external entities while parsing.
- · Properties
  - Improved behavior for strings with no value.
- Android resources
  - Improved newlines handling.
  - Strip leading and trailing whitespace.
- PHP
  - Added support for nested named arrays and nested unnamed arrays.
- ts
- Handle gracefully empty location tag.
- Encode *po2ts* output as UTF-8.
- resx
  - Improved skeleton.
  - Fixed indent of the </data> elements.

## Languages

• Added Kabyle language.

#### **API changes**

- Added functions to retrieve language and country ISO names.
- If available, *pycountry* is used first to get language names translations.

#### General

- Python 3 fixes
- · Added more tests

... and loads of general code cleanups and of course many many bugfixes.

#### **Contributors**

This release was made possible by the following people:

Dwayne Bailey, Michal Čihař, Taras Semenenko, Leandro Regueiro, Rimas Kudelis, BhaaL, Muend Belqasem, Jens Petersen.

And to all our bug finders and testers, a Very BIG Thank You.

#### **Translate Toolkit 2.1.0**

Released on 17 March 2017

This release contains many improvements and bug fixes.

#### **Major changes**

- Fixed RomanianChecker checks.
- Added an iOS checker style.
- Changed plural equations for Slovenian, Persian, Kazakh and Kyrgyz.
- Several fixes in formats and tools.

## **Detailed changes**

#### **Python 3 support**

• Python 3.6 is now supported.

#### Requirements

- Updated and pinned requirements.
- Now recommended requirements pulls required requirements.

#### **Formats and Converters**

- · All formats
  - locationindex now uses first duplicate unit in case of several units having the same location in order to keep duplicate entries in some formats when converting from PO format.
- PO
  - Only add duplicate unit if *msgcxt* is unique, in order to be able to convert monolingual formats with duplicate entries to PO.
- Properties
  - Added support for Joomla dialect.
- ts
- Set the right context on the units.

- YAML
  - Fixed parsing of unicode values in lists.
- HTML
  - Use character offset in line for unit location in order to keep parsing repeated strings in different units.
- txt
  - Use line number on unit location to keep parsing repeated strings in different units.

#### **Filters and Checks**

- Fixed RomanianChecker checks.
- Added an iOS checker style to detect iOS variables styles such as %@ and \$ (VAR).

#### **Tools**

• posegment no longer outputs duplicate headers,

#### Languages

• Changed plural equations for Slovenian, Persian, Kazakh and Kyrgyz.

## **API changes**

• Changed management of Xapian locks to prevent database corruption.

#### General

- Python 3 fixes
- Removed unused code

 $\ldots$  and loads of general code cleanups and of course many many bugfixes.

#### **Contributors**

This release was made possible by the following people:

Dwayne Bailey, Leandro Regueiro, Michal Čihař, Ryan Northey, Friedel Wolff, Olly Betts, Claude Paroz.

And to all our bug finders and testers, a Very BIG Thank You.

#### **Translate Toolkit 2.0.0**

Released on 27 January 2017

This release contains many improvements and bug fixes. While it contains many general improvements, it also specifically contains needed changes and optimizations for the upcoming Pootle 2.8.0 and Virtaal releases.

#### **Major changes**

- Python 3 compatibility thanks to Claude Paroz
- Dropped support for Python 2.6
- Support for new 120n format
- Translate Toolkit can now easily be installed on Windows
- · Changes in storage API to expose a more standardized API

## **Detailed changes**

## **Python 3 support**

- Translate Toolkit went through a massive code cleanup looking forward Python 3 compatibility. There might be quirks that need to be fixed, so please test and report any issue you might find.
- Python 3.3-3.5 is now supported.

#### Requirements

- 1xml requirement was raised to 3.5.0 in order to simplify code.
- Updated and pinned requirements
- Removed misleading extra requirements files

#### **Formats and Converters**

- PO
  - msgid comments (KDE style) are only parsed from msgid now.
  - Fixed parsing of PO files with first entry in unicode
  - Fixed parsing of locations with percent char
- XLIFF
  - Unaccepted ASCII control characters are now escaped in XLIFF
- DTD
  - Newlines are now skipped when parsing (issue 3390).
  - Invalid ampersands are not scrubbed anymore.
  - label+accesskey is now only extracted if it is not followed by space.
- Properties
  - Keys can contain delimiters if they are properly wrapped (issue 3275).
  - Fix control characters escaping for utf-8 encoding.
  - po2prop removes fully untranslated units if required
  - *po2prop* skips first entry in PO file (issue 3463)

- · Mozilla .lang
  - {ok} marker is now more cleanly removed
  - Always output last unit followed by trailing newlines
  - Added support for headers and tag comments
  - MAX LENGTH is now parsed into comment
  - File line endings are now remembered, defaulting to Unix LF
- · Mozilla's 120n
  - Added this new format storage class
  - Added variants and traits support
  - Added new converters 120n2po and po2120n
- · Android
  - Unknown locales no longer produce failures.
  - Simplify newlines handling as the format now handles n and newline equally (issue 3262)
  - Moved all namespaces to <resources> element.
  - Simplified newlines handling
- ODF
  - *odf2xliff* now extracts all the text (issue 3239).
- ts
- XML declaration is written with double quotes.
- Self-closing for 'location' elements are not output anymore.
- JSON
  - Output now includes a trailing newline.
  - Unit ordering is maintained (issue 3394).
  - Added --removeuntranslated option to po2json
- YAML
  - YAML format support has been added.
- txt
  - po2txt works correctly again when --threshold option is passed (issue 3499)
- ical
  - Enabled this format for Python 3 too.
- TermBase eXchange (TBX)
  - tbx2po converter added
  - Added basic support for Parts of Speech and term definitions.
- Fixed error when writing back to the same file (issue 3419).

#### **Filters and Checks**

- · Added the ability to skip some checks for some languages in specific checkers
- accelerators check reports an error if accelerator is present for several Indic languages in MozillaChecker checker.
- Added 120nChecker to do custom checking for Mozilla's new 120n format.
- LibreOffice checker no longer checks for Python brace format (issue 3303).
- LibreOffice validxml check correctly matches self-closing tags.
- Numbers check now handles non latin numbers. Support for non latin numbers has been added for Arabic, Assamese, Bengali and Persian languages.
- Fixed issue that prevented standard checks from being used in Pootle with default settings.
- Fixed missing attribute warning displayed when using GnomeChecker, LibreOfficeChecker and MozillaChecker checkers.
- Added language specific RomanianChecker.

#### **Tools**

- posegment now correctly segments Japanese strings with half width punctuation sign (issue 3280).
- pocount now outputs csv header in one line. It also outputs using color.
- buildxpi was adjusted to current Mozilla needs

#### Languages

- Fixed plural form for Montenegro, Macedonian, Songhay, Tajik, Slovenian and Turkish.
- Added plural forms for Bengali (Bangladesh), Konkani, Kashmiri, Sanskrit, Silesian and Yue (Cantonese).
- · Added valid accelerators for Polish.
- · Renamed Oriya to Odia.
- Altered Manipuri name to include its most common name Meithei.
- Added language settings for Brazilian Portuguese.
- Added Danish valid accelerators characters (issue 3487).
- Added additional special characters for Scottish Gaelic.

#### Setup

- Fixed Inno Setup builds allowing to easily install Translate Toolkit on Windows using the pip installer. Commands are compiled to .exe files.
- · Updated installation instructions for Windows

#### **API** changes

- Dropped translate.misc.dictutils.ordereddict in favor of collections.OrderedDict.
- Added encoding handling in base TranslationStore class exposing a single API.
- Encoding detection in TranslationStore has been improved.
- Standardized UnitClass definition across TranslationStore subclasses.
- translate.misc.multistring.multistring:
  - Fixed list coercion to text
  - Fixed comparison regression with multistrings (issue 3404).
  - Re-added str method (issue 3428).
  - Fixed \_\_hash\_\_ (issue 3434).

#### **API** deprecation

- Passing non-ASCII bytes to the multistring class has been deprecated, as well as the encoding argument to it. Applications should always construct multistring objects by passing characters (unicode in Python 2, str in Python 3), not bytes. Support for passing non-ASCII bytes will be removed in the next version.
- TxtFile.getoutput() and dtdfile.getoutput() have been deprecated. Either call bytes(<file\_instance>) or use the file\_instance.serialize() API if you need to get the serialized store content of a TxtFile or dtdfile instance.

#### General

- Dropped support for Python 2.6 since it is no longer supported by the Python Foundation. Sticking to it was making us difficult to maintain code while we move to Python 3.
- Misc docs cleanups.
- · Added more tests.
- · Increased Python code health.
- Legacy, deprecated and unused code cleansing:
  - Dropped code for no longer supported Python versions.
  - Removed unused code from various places across codebase.
  - The legacy translate.search.indexing.PyLuceneIndexer1 was removed.
  - The deprecated translate.storage.properties.find\_delimiter() was removed and replaced by the translate.storage.properties.Dialect.find\_delimiter() class method.
  - Python scripts are now available via console\_scripts entry point, thus allowing to drop dummy files for exposing the scripts.

... and loads of general code cleanups and of course many many bugfixes.

#### **Contributors**

This release was made possible by the following people:

Claude Paroz, Leandro Regueiro, Dwayne Bailey, Michal Čihař, Taras Semenenko, Ryan Northey, Stuart Prescott, Kai Pastor, Julen Ruiz Aizpuru, Friedel Wolff, Hiroshi Miura, Thorbjørn Lindeijer, Melvi Ts, Jobava, Jerome Leclanche, Jakub Wilk, Adhika Setya Pramudita, Zibi Braniecki, Zdenek Juran, Yann Diorcet, Nick Shaforostoff, Jaka Kranjc, Christian Lohmaier, beernarrd.

And to all our bug finders and testers, a Very BIG Thank You.

#### **Translate Toolkit 1.13.0**

Released on 13 May 2015

This release contains many improvements and bug fixes. While it contains many general improvements, it also specifically contains needed changes for the upcoming Pootle 2.6.0 and 2.7.0 and Virtual releases.

It is just over 6 months since the last release and there are many improvements across the board. A number of people contributed to this release and we've tried to credit them wherever possible (sorry if somehow we missed you).

#### **Highlighted improvements**

#### **Major changes**

- · New converters for IDML format
- Support for new .Net Resource (.resx) format
- Extensive cleanup on ODF converters
- · New quality checks

#### **Formats and Converters**

- IDML
  - Added the idml2po and po2idml converters.
- .Net Resource (.resx)
  - Added store to represent the format and the resx2po and po2resx converters.
- Android
  - Improved escape and unescape of Android resources with HTML markup.
  - Fixed bug in canceling whitespaces with backslash when unescaping.
- ODF
  - Removed the --engine option in odf2xliff because the itools third party library is no longer used.
- TS
  - Pretty print output the same as Qt Linguist (issue 1420)
- JSON

- Dump content on memory instead of copy of parsed file (issue 3249).
- PHP
  - Correctly roundtrip PHP with spaces after array (issue 3231).
- · Mozilla lang
  - Import only real comments (starting with #), not meta tags (starting with ##).
- XLIFF
  - Mark units as needing attention if sources don't match when merging units.
  - pot2po now also accepts files with .xliff extension
- po2moz
  - Fixed handling of files with fullstop in filename

## **Quality Checks**

- Added quality check for Python brace format.
- Added the ability to skip some quality checks for the he, ug, zh\_CN, zh\_HK and zh\_TW languages.
- Expanded printf quality check to support reordering boost::format positional directives.
- Expanded docstrings to include fully detailed descriptions in order to display them on Pootle.

#### **Tools**

• Removed the unnecessary dependency on lxml in pretranslate (issue 1909)

#### Languages

- Language plurals:
  - Fixed plural forms for ga and pt\_BR languages
  - Added new plural forms for new languages
- · Adjusted punctuation for zh
- · Corrected "Songhay" language name

#### General

- Fixed bug in file discovery that prevented Pootle Pootle's terminology feature from working properly in some scenarios.
- Docs:
  - Major rewrite of releasing instructions
  - Reorganized string-related guidelines on styleguide
  - Other minor docs cleanups
- ODF code extensive cleanups:

- Applied tons of PEP8 and style guide cleanups
- Removed unused code
- Removed unused test ODT file
- Added lots of docstrings
- Simplified code to ease maintainability and improve readability
- Dropped no longer working automatic publishing in PyPI and SourceForge
- · Several changes to speed up Travis builds
- · Unhid some tests

... and loads of general code cleanups and of course many many bugfixes.

#### **Contributors**

This release was made possible by the following people:

Leandro Regueiro, Dwayne Bailey, Yaron Shahrabani, Sarah Hale, Sietse Brouwer, Jerome Leclanche, Julen Ruiz Aizpuru, Michael Andres, William Grzybowski, SirAnthony, Rafael Ferreira, Luka Kama, Francesco Lodolo, Buganini, babycaseny.

And to all our bug finders and testers, a Very BIG Thank You.

#### **Translate Toolkit 1.12.0**

Released on 12 August 2014

This release contains many improvements and bug fixes. While it contains many general improvements, it also specifically contains needed changes and optimizations for the upcoming Pootle 2.6.0 and Virtual releases.

It is just over 6 months since the last release and there are many improvements across the board. A number of people contributed to this release and we've tried to credit them wherever possible (sorry if somehow we missed you).

#### **Highlighted improvements**

## Major changes

- Properties and DTD formats fix a number of issues
- Massive code cleanup looking forward Python 3 compatibility
- Important changes in development process to ease testing

#### **Formats and Converters**

- Mozilla properties
  - If a unit has an associated access key entry then these are combined into a single unit
  - Encoding errors are now reported early to prevent them being masked by subsequent errors
  - Leading and trailing spaces are escaped in order to avoid losing them when using the converters
  - The \uNN characters are now properly handled

- po2prop Now uses the source language accesskey if translation is missing
- Fixed conversion of successive Gaia plural units in prop2po

#### • DTD

- The & amp; entity is automatically expanded when reading DTD files, and escaped back when writing them
- Underscore character is now a valid character in entity names
- Nonentities at end of string are now correctly handled
- po2dtd:
  - \* Now uses the source language accesskey if target accesskey is missing
  - \* Doesn't remove stray & as they probably & amp;

#### • HTML

- The HTML5 figcaption tag is now localizable
- The title attribute is now localizable
- po2html now retains the untranslated attributes

#### · Accesskeys

- Now accesskeys are combined using the correct case
- Added support for accesskey after ampersand and space

#### • PHP

- Fall back to default dialect after adding every new unit
- Added support for empty array declaration when it is filled later

#### · Android

- Added support for plurals
- Text is now properly escaped when using markup

## • Qt Linguist (.ts)

- The message id attribute is added to contextname
- Files now output the XML declaration (issue 3198)

#### • RC

- RC format received some bugfixes and now ignores TEXTINCLUDE sections and one line comments (//)

#### XLIFF

- xliff2po now supports files with .xliff extension

#### · OS X .strings

- Added support for UTF-8 encoded OS X strings

#### Testing

- Added new tests for the UTF-8 encoded OS X strings, Qt linguist and RC formats and the rc2po converter

#### **Version Control improvements**

· Added support for Subversion . svn directories

#### Checks

• Added specific checks for LibreOffice

#### **Tools**

• The pocount tool has now a better counting algorithm for things that look like XML

## Mozilla tooling fixes

- · Added support to check for bad accesskeys in .properties files
- Now the Mozilla roundtrip script can be silently run
- · Added a new Gaia roundtrip script
- The buildxpi --disable-compile-environment option has been restored, resulting in huge speed improvements

#### General

- Extensive cleanup of setup script
- Some bugfixes for placeables
- · Misc docs cleanups
- Recovered diff-match-patch to provide support for old third party consumers
- Minor change in placeables to correctly insert at an existing parent if appropriate
- Code cleanups:
  - Applied tons of PEP8 and style guide cleanups
  - Python 2.6 is our new minimum:
    - \* Removed lots of code used to support old Python versions
    - \* Dropped custom code in favor of Python standard libraries
    - \* Updated codebase to use newer libraries
    - \* Changed code to use newer syntax seeking Python 3 compatibility
  - Updated some third party bundled software: CherryPy, BeautifulSoup4
  - Added document to track licenses used by third party bundled code
  - Removed TODO items. Some of them were moved to the bug tracker
- Development process:
  - Added a functional test framework
  - Added dozens of new unit and functional tests

- Expanded the tasks performed in Travis: pep8, pytest-xdist, compile all files, coveralls.io, . . .

... and loads of general code cleanups and of course many many bugfixes.

#### **Contributors**

This release was made possible by the following people:

Dwayne Bailey, Jerome Leclanche, Leandro Regueiro, Khaled Hosny, Javier Alfonso, Friedel Wolff, Michal Čihař, Heiki Ojasild, Julen Ruiz Aizpuru, Florian Preinstorfer, damian.golda, Zolnai Tamás, Vladimir Rusinov, Stuart Prescott, Luca De Petrillo, Kevin KIN-FOO, Henrik Saari, Dominic König.

And to all our bug finders and testers, a Very BIG Thank You.

#### **Translate Toolkit 1.11.0**

Released on 22 January 2014

This release contains many improvements and bug fixes. While it contains many general improvements, it also specifically contains needed changes and optimizations for the upcoming Pootle 2.5.1 and Virtaal releases.

It is just over a ten months since the last release and there are many improvements across the board. A number of people contributed to this release and we've tried to credit them wherever possible (sorry if somehow we missed you).

#### **Highlighted improvements**

#### **Major changes**

- The PO format now matches Gettext more closely
- PHP format adds a number of new features
- Support for Python 2.5 has been dropped

#### **Formats and Converters**

- Gettext PO:
  - cPO now handles fuzzy obsolete messages correctly
  - Line wrapping improvement allow PO files to more closely match Gettext
  - Optimization to increase performance
- PHP:
  - Warn about duplicate entries
  - Allow blank spaces in array declaration (issue 2646)
  - Support nested arrays (issue 2240)
- XLIFF:
  - Correctly parse XLIFF 1.2
- Properties
  - Blank source text is now always translated

- Fuzzy units are discarded with -remove-untranslated
- prop2po no longer drops entries that are translated the same as the source
- TMX:
  - po2tmx support comments
- Android:
  - Detect untranslatable resource strings
  - Various format improvements
- HTML:
  - Output HTML source in po2html when a unit is fuzzy (issue 3145)
- New conversion options:
  - --timestamp skip conversion if the output file has a newer timestamp (Makefile-alike)
  - --threshold in po2\* converters this allows you to specify a percentage complete threshold. If the PO files passes this threshold then the file is output (issue 2998)
  - --removeuntranslated Extend this option to po2dtd and thus po2moz don't output untranslated text (issue 1718)

# Language specific fixes

- The toolkit now supports: Sakha, N'ko, Turkish, improvements for Bengali & Hindi
- · Pootle special characters are now stored on Toolkit and available for other tools to use
- Rules for language ab are now available for language ab\_CD

# Checks

- Spelling test improvements including speed and optimization
- Reduce false positive for the filepath test in cases of self closing tags e.g. <br/> />
- Lowered the accelerator check severity to reduce false positive impact

# Mozilla tooling fixes

- Better decoding of some characters in DTD e.g » and &x0022 (")
- .lang Improved support for untranslated entries
- buildxpi:
  - Can now build multiple languages at once (issue 2999)
  - Set a max product version to allow the language pack to continue to work once the browser version has moved out of Aurora channel
- Dropped native XPI building support (untested and no longer used)
- Add Mozilla plural formulas, in time we'll handle Mozilla plurals correctly

#### General

- Dropped support for Python 2.5 since it is no longer supported by the Python Foundation. Also sticking to it was preventing us from using features that are not supported on Python 2.5 but they are on later versions.
- Dropped psyco support it is no longer maintained
- Use logging throughout instead of sys.stderr
- Lots of cleanups on docs: TBX, PHP, added Android and JSON docs
- Use requirements files for documenting all requirements and make it easy to install Translate Toolkit using pip
- · Added some functional tests
- · Improve searching to find words with hyphens
- Choose the closest repo in nested VCS
- Test suite down to zero failing tests
- Handle a broken version of python-Levenshtein
- Improve handling of escapes in wrapping

... and loads of internal changes to improve maintainability, remove unused imports, remove unused code and general code cleanups, some changes to ensure future Python 3 portability and of course many many bugfixes.

#### **Contributors**

This release was made possible by the following people:

Dwayne Bailey, Leandro Regueiro, Alexander Dupuy, Friedel Wolff, Khaled Hosny, Michal Čihař, Jordi Mas, Stuart Prescott, Trung Ngo, Ronald Sterckx, Rail Aliev, Michael Schlenker, Martin-Zack Mekkaoui, Iskren Chernev, Luiz Fernando Ranghetti & Christian Hitz

And to all our bug finders and testers, a Very BIG Thank You.

# **Translate Toolkit 1.10.0**

Released on 12 March 2013

This release contains many improvements and bug fixes. While it contains many general improvements, it also specifically contains needed changes for the upcoming Pootle 2.5.0.

It is just over a year since the last release so there are many improvements across the board. A number of people contributed to this release and we've tried to credit them wherever possible (sorry if somehow we missed you).

# **Highlighted improvements**

- · Android format support
- Version control improvements
- · Source now on Github all our code is now on github
- Documentation migrated all from our wiki into the code and Read The Docs
- Continuous Integration using Travis

# Most important for Pootle

- Version control improvements
- Categorize pofilter checks into critical, functional, cosmetic, etc

#### **Formats and Converters**

- Android format support [Michal Čihař]
- Mozilla .lang, many improvements
- PHP support for defintions, // comments and improved whitespace preservation
- PO: X-Merge-On header to explicitly demand a conversion strategy instead of guessing
- .properties: BOMs in messages and C style comments [Roman Imankulov]
- Mac OS String formatting improved [Roman Imankulov]
- The spaces in DTD files are now preserved. For example the spaces in <!ENTITY some.label "definition"> around the entity name some.label are now kept.
- The matching criterion when merging units can now be specified with the X-Merge-On header. Available values for this header are *location* and *id*. By default merges will be done by matching IDs. This supersedes the effects of the X-Accelerator header when merging and establishes an explicit way to set the desired matching criterion.

# **Version Control improvements**

- Interface for adding files to a repository & Implement .add() for all VCSs.
- Caching of VC version info
- Don't look for VCS if it's not available
- Stop looking for VCS at a given parent
- Subversion VC tests
- Alway pass -m to 'commit' in Subversion to prevent blocking

#### Checks

- New OpenOffice variables style used in extensions
- Check for self-closing tags in the xmltags test [Seb M].
- · GConf test fixes
- Terminology checker type for future terminology features
- Categorize pofilter checks into critical, functional, cosmetic, etc
- Added support for Objective-C %@ printf specifiers

# Language specific fixes

- Correct plurals: Scottish Gaelic (gd), Irish
- Plural rules: Fulah, Brazilian Portuguese
- Punctuation rules and tests to ignore for: Burmese, Urdu, Afrikaans, Wolof

#### **Documentation**

- Moved to Git and we are now using reStructured Text and Sphinx
- Published in Read The Docs (RTD).
- Old wiki migrated to RTD.
- · New clean theme for documentation and website
- API and code epydoc moved to reStructured Text.
- Translate code Style Guide written

# Mozilla tooling fixes

- Mozilla specific test for dialog size settings
- · Gaia properties dialect and plural handling
- Fixes and improvement to the Firefox build scripts
- · Improved accesskey detection
- Improved DTD escaping for &quote, %, etc
- Improvement of DTD to align with Base classes
- Support new  $\{\{xx\}\}\$  variable style introduced in PDF viewer

... and refactoring, PEP8, test coverage and of course many many bugfixes.

#### **Contributors**

This release was made possible by the following people:

Dwayne Bailey, Friedel Wolff, Leandro Regueiro, Julen Ruiz Aizpuru, Michal Čihař, Roman Imankulov, Alexander Dupuy, Frank Tetzel, Luiz Fernando Ranghetti, Laurette Pretorius, Jiro Matsuzawa, Henrik Saari, Luca De Petrillo, Khaled Hosny, Dave Dash & Chris Oelmueller.

And to all our bug finders and testers, a Very BIG Thank You.

#### **Translate Toolkit 1.9.0 Released**

Released on 12 April 2011

This release contains many improvements and bug fixes. While it contains many general improvements, it also specifically contains needed changes for the upcoming Pootle 2.1.6 and Virtaal 0.7.

# **Highlighted improvements**

- · Faster terminology matching
- Several small optimisations to performance and memory use
- More advanced state support (visible in pocount and Virtaal 0.7)
- Improved language detection models (+South African languages)
- Improve handling of printf variable reordering [Jacques Beaurain]
- Review of the wording of the messages of pofilter checks
- Better sentence segmentation for some non-Latin languages
- More supported formats for podebug
- Extra options for pomerge, pogrep and po2oo/xliff2oo.

The new pogrep options made this possible for GNOME.

#### Most relevant for Pootle

- Support for Xapian 1.2 (issue 1766) [Rimas Kudelis]
- Work around some changes introduced in Django 1.2.5/1.3

# Format support

- Always use UNIX line endings for PO (even on Windows)
- XLIFF and .ts files now shows "fuzzy" only the target present
- Improved support for .ts comment as context (issue 1739)
- Support for Java properties in UTF-8 encoding
- More natural string ordering in json converter
- Improved handling of trailing spaces in Mozilla DTD files
- Removed unused support for \_old\_ KDE plurals in pocount

... and several small bugfixes

# **Translate Toolkit 1.8.1**

Released on 19 Novermber 2010

Today the Translate team released version 1.8.1 of the Translate Toolkit. The Translate Toolkit contains many useful tools for translation, management, and quality control. It is the technology platform for Pootle, Virtaal, and other software.

This release contains many improvements and bug fixes. It is a recommended upgrade for users of Pootle and Virtaal. There were over 200 commits since version 1.8.0.

# This work was made possible by volunteers and our funders:

• ANLoc, funded by IDRC

# **Highlighted improvements**

#### File formats:

- A rewrite and major improvement of the html format and html2po converter
- · New JSON format introduced
- Support for Universal Terminology Exchange (UTX) format
- Support for Java properties files encoded in UTF-8
- Improvements to CSV format, and improved compatibility with Excel exports
- · Bug fixes to Qt .ts
- Support for XLIFF's state attributes (pocount now lists detailed state statistics)
- · Minor bug fixes for PHP format

#### Languages and quality checks:

- Support for Persian quotations
- · Major performance improvements to quality checks

Pootle will regenerate all statistics with the new Translate Toolkit installed. Read about the quality checks.

# Other improvements:

- Improvements to stability of Lucene text indexing (affecting Pootle)
- Parameter for po2prop to ignore untranslated strings
- Many improvements to pot2po including Qt ts support, improved handling of extra XML namespaces in XLIFF, and performance improvements.

# Further resources:

- · Full feature list
- · Download
- Bugs

Happy translating!

The Translate team

#### 3.1.2 Historic releases

# **Translate Toolkit 1.8.0**

Released on 17 August 2010

Release notes for this release are still to be recovered, feel free to do some way back searching and create a Pull Request for this release.

# **Translate Toolkit 1.7.0**

Released on 13 May 2010

Release notes for this release are still to be recovered, feel free to do some way back searching and create a Pull Request for this release.

# **Translate Toolkit 1.6.0**

Released on 3 March 2010

# PO files now always have headers

Generated PO files now always contain headers. This will mainly affect the output of pofilter and pogrep. This should allow better interoperability with gettext tools, and allowed for some improvement in the code. You should still be able to use headerless files in msgmerge, although it is recommended that PO files are consistently handled with headers wherever possible.

# **Translate Toolkit 1.5.3**

Released on 4 February 2010

Release notes for this release are still to be recovered, feel free to do some way back searching and create a Pull Request for this release.

#### **Translate Toolkit 1.5.2**

Released on 13 January 2010

Release notes for this release are still to be recovered, feel free to do some way back searching and create a Pull Request for this release.

#### **Translate Toolkit 1.5.1**

Released on 8 December 2009

Release notes for this release are still to be recovered, feel free to do some way back searching and create a Pull Request for this release.

# **Translate Toolkit 1.5.0**

Released on 25 November 2009

Release notes for this release are still to be recovered, feel free to do some way back searching and create a Pull Request for this release.

# **Translate Toolkit 1.4.1**

Released on 6 November 2009

#### **CSV** column header names

The names given to CSV column headers have been changed. Early releases of *csv2po* would name the columns "comment,original,translation". This was done mostly to make it easy for non-technical translators. However, comments in the command line help used terms like source and target. This release changes the column header names to "location,source,target", this aligns with terms used throughout the toolkit.

If you have CSV file generated by older versions of the toolkit then a header entry of "comment,original,translation" will be turned into a unit instead of being ignored. You can either change your CSV file to use the headers "location,source,target" or delete the header row completely. Once this is done the files will work as expected.

#### **Translate Toolkit 1.4.0**

Released on 27 August 2009

# Java and Mozilla .properties

Unusual keys, separators and spacing should all be handled correctly now. Some Mozilla .properties files might now have changed. Regenerate your Mozilla 110n files from fresh POT files without any changes to your PO files to ensure that you can see and review these changes.

# Hashing in podebug

The --hash option in *podebug* has been replaced by a format specifier %h to be able to better control the positioning of the hash value.

#### **Translate Toolkit 1.3.0**

Released on 11 February 2009

Several duplicate styles were removed as has been warned about long before. Please check the recommendations posted at the time that msgctxt was added on how to migrate.

#### **Translate Toolkit 1.2.1**

Released on 29 December 2008

Release notes for this release are still to be recovered, feel free to do some way back searching and create a Pull Request for this release.

#### **Translate Toolkit 1.2.0**

Released on 24 November 2008

#### **New formats**

The toolkit now supports:

- Qt Phrase Book (.qph)
- Qt.ts v1.1

This allows reading, counting and working on these formats. The *ts2po* converter has not been changed so you will not be able to benefit from the new .ts support. However, you can use the format for translation memory, etc as its is now fully base class compliant.

# Stats database change

There were some changes in the database used by pocount for storing statistics. The location of the database might also have changed, depending on what the last version is that you used. Remove the file stats.db from any of ~/.translate\_toolkit, ~/.wordforge (or the corresponding directories on your Windows installation.

#### Valid accelerators

The *pofilter* accelerator test is now able to make use of a list of valid accelerators. This allows translators to control the behaviour of the test for their language and add or remove characters that can be used as accelerators. Please define the valid accelerators for your language and these will then be included in future releases of the toolkit. By default the old process is followed so that if you take no action then this check will continue to work as expected.

#### branches

These are branches that contain quite invasive changes that will most likely be merged into the main development and be released sometime in the future.

# toolkit-C-po

Converting the current Python based PO parser to the Gettext C based parser for PO. This offers quite a dramatic speed improvement and conformance to the output found in Gettext itself. For most users there will be a number of changes in layout of the files as they will now conform fully to Gettext layout. The 'keep' option in --duplicatestyle will no longer be supported as this is not valid Gettext output.

#### Translate Toolkit 1.1.1

Released on 2 April 2008

#### Premature termination of DTD entities

Although this does not occur frequently a case emerged where some DTD entities where not fully extracted from the DTD source. This was fixed in issue 331.

We expect this change to create a few new fuzzy entries. There is no action required from the user as the next update of your PO files will bring the correct text into your translations, if you are using a translation memory your translation might be recovered from obsolete translations.

#### **Translate Toolkit 1.1.0**

Released on 22 January 2008

# oo2po Help (helpcontent2) escaping fixed

OpenOffice.org Help (helpcontent2) has notoriously contained some unreadable esacping, e.g. \\\\<tag attr=\\"value\\"\\\>. The escaping has been fixed and oo2po now understands helpcontent2 escaping while leaving the current GUI escape handling unaltered.

If you have not translated helpcontent2 then you are unaffected by this change. If you have translated this content then you will need to follow these instructions when upgrading.

If you follow normal procedures of creating POT files and upgrading your PO files using pot2po then your strings will not match and you will obtain files with many fuzzies. To avoid this do the following:

- 1. Make sure your PO files contain no fuzzy entries
- 2. Use po2oo from the previous release to create and SDF file
- 3. Upgrade to the latest Translate Toolkit with new po2oo
- 4. Use po200 -1 xx-YY your.sdf po to create a new set of PO files with correct escaping

You can choose to do this with only your helpcontent2 PO files if needed, this will allow you to leave your GUI work in its current state. Simply do the above procedure and discard all PO files except helpcontent2, then move these new helpcontent2 files into your current work.

# prop2po uses developer comments

prop2po used to place comments found in the source .properties file in traditional translator comments, they should of course go into developer comments. The reason for this change is twofold, it allows these comments to be correctly managed and it is part of the process of cleaning up these formats so that they are closer to the base class and can thus work with XLIFF.

For the user there will be fairly large changes as one comment format moves to the next. It is best to *cleanup translator comments* and get your translations into a fit state, i.e. no fuzzies, and then proceed with any migrations.

#### moz2po no longer uses KDE comments

moz2po has traditionally used KDE style comments for storing comments aimed at translators. Many translators confuse these and try to translate them. Thus these have been moved into automatic or developer comments. The result for many people migrating Mozilla PO files will be that many strings will become fuzzy, you can avoid much of this by using pot2po which should intelligently be able to match without considering the KDE comments.

The best strategy is to get your translations into a relatively good shape before migration. You can then migrate them first to a new set of POT files generated from the same source files that the translation is based on. Eliminate all fuzzies as these should only relate to the changes in layout. Then proceed to migrate to a new set of POT files. If you cannot work against the original source files then the best would be to also first eliminate fuzzy matches before proceeding to translation. Your fuzzies will include changes in layout and changes in content so proceed carefully.

At the end of this you should have PO files that conform to the Gettext standard without KDE comments.

#### **Read and Write MO files**

You can read and write Gettext MO files (compiled PO files). Thus pocount can now count files on your filesystem and you can also compile MO files using pocompile. MO files can be compiled from either PO or XLIFF sources.

MO will now also produce correct output for msgctxt and plural forms found in PO files.

# Read Qt .qm files

We can now read Qt .qm files, thus pocount can count the contents of compiled files. We cannot however write .qm files at this time.

#### **Translate Toolkit 1.0.1**

Released on 23 June 2007

# pot2po will create new empty PO files if needed

From version 1.0.1, pot2po will create empty PO files corresponding to new POT files that might have been introduced. If some new POT files are present in the input to pot2po, you will see a new PO file appear in your output directory that was not in your old PO files. You will not lose any data but in the worst case you will see new files on projects that you thought were fully translated.

#### **Translate Toolkit 1.0**

Released on 1 June 2007

# Improved XLIFF support

Many toolkit tools that only worked with PO files before, can now also work with XLIFF files. pogrep, pocount, pomerge, and pofilter all work with XLIFF, for example.

# **Pretty XML output**

All XML formats should now be more human readable, and the converters to Qt .ts files should work correctly again.

# Fuzzy matching in pot2po is optional

Fuzzy matching can now be entirely disabled in *pot2po* with the --nofuzzymatching parameter. This should make it much faster, although pot2po is **substantially** faster than earlier versions, especially if *python-Levenshtein* is installed.

#### Old match/Levenshtein.py\* can cause name clash

The file previously called match/Levenshtein.py was renamed to lshtein.py in order to use the python-Levenshtein package mentioned above. If you follow the basic installation instructions, the old file will not be overwritten, and can cause problems. Ensure that you remove all files starting with Levenshtein.py in the installation path of the translate toolkit, usually something like /usr/lib/python2.4/site-packages/translate/search/. It could be up to three files.

# PO file layout now follows Gettext more closely

The toolkits output PO format should now resemble Gettext PO files more closely. Long lines are wrapped correctly, messages with long initial lines will start with a 'msgid "" entry. The reason for this change is to ensure that differences in files relate to content change not format change, no matter what tool you use.

To understand the problem more clearly. If a user creates POT files with e.g. oo2po. She then edits them in a PO editor or manipulate them with the Gettext tools. The layout of the file after manipulation was often different from the original produced by the Toolkit. Thus making it hard to tell what where content changes as opposed to layout changes.

The changes will affect you as follows:

- 1. They will only impact you when using the Toolkit tools.
- 2. You manipulate your files with a tool that follows Gettext PO layout
  - your experience should now improve as the new PO files will align with your existing files
  - · updates should now only include real content changes not layout changes
- 3. You manipulate your files using Toolkit related tools or manual editing
  - your files will go through a re-layout the first time you use any of the tools
  - · subsequent usage should continue as normal
  - any manipulation using Gettext tools will leave your files correctly laid out.

Our suggestion is that if you are about to suffer a major reflow that your initial merge contain only reflow and update changes. Do content changes in subsequent steps. Once you have gone through the reflow you should see no layout changes and only content changes.

# Language awareness

The toolkit is gradually becoming more aware of the differences between languages. Currently this mostly affects pofilter checks (and therefore also Pootle) where tests involving punctuation and capitalisation will be more aware of the differences between English and some other languages. Provisional customisation for the following languages are in place and we will welcome more work on the language module: Amharic, Arabic, Greek, Persian, French, Armenian, Japanese, Khmer, Vietnamese, all types of Chinese.

# New pofilter tests: newlines and tabs

The escapes test has been refined with two new tests, newlines and tabs. This makes identifying the errors easier and makes it easier to control the results of the tests. You shouldn't have to change your testing behaviour in any way.

# Merging can change fuzzy status

pomerge now handles fuzzy states:

```
pomerge -t old -i merge -o new
```

Messages that are fuzzy in *merge* will now also be fuzzy in *new*. Similarly if a fuzzy state is present in *old* but removed in *merge* then the message in *new* will not be fuzzy.

Previously no fuzzy states were changed during a merge.

#### pofilter will make Mozilla accelerators a serious failure

If you use *pofilter* with the --mozilla option then accelerator failures will produce a serious filter error, i.e. the message will be marked as fuzzy. This has been done because accelerator problems in your translations have the potential to break Mozilla applications.

# po2prop can output Mozilla or Java style properties

We have added the --personality option to allow a user to select output in either java, or mozilla style (Java property files use escaped Unicode, while Mozilla uses actual Unicode characters). This functionality was always available but was not exposed to the user and we always defaulted to the Mozilla style.

When using *po2moz* the behaviour is not changed for the user as the programs will ensure that the properties convertor uses Mozilla style.

However, when using *po2prop* the default style is now java, thus if you are converting a single .properties file as part of a Mozilla conversion you will need to add --personality=mozilla to your conversion. Thus:

```
po2prop -t moz.properties moz.properties.po my-moz.properties
```

#### Would become:

```
po2prop --personality=mozilla -t moz.properties moz.properties.po my-moz.properties
```

Note: Output in java style escaped Unicode will still be usable by Mozilla but will be harder to read.

# Support for compressed files

There is some initial support for reading from and writing to compressed files. Single files compressed with gzip or bzip2 compression is supported, but not tarballs. Most tools don't support it, but pocount and the --tm parameter to pot2po will work with it, for example. Naturally it is slower than working with uncompressed files. Hopefully more tools can support it in future.

#### **Translate Toolkit 0.11**

Released on 24 March 2007

# po2oo defaults to not check for errors

In po200 we made the default --filteraction=none i.e. do nothing and don't warn. Until we have a way of clearly marking false positives we'll have to disable this functionality as there is no way to quiet the output or mark non errors. Also renamed exclude to exclude-all so that it is clearer what it does i.e. it excludes 'all' vs excludes 'serious'.

# pofilter xmltags produces less false positives

In the xmltags check we handle the case where we had some false positives. E.g. "<Error>" which looks like XML/HTML but should actually be translated. These are handled by

- 1. identifying them as being the same length as the source text,
- 2. not containing any '=' sign. Thus the following would not be detected by this hack. "An <Error> occurred" -> "<Error name="bob">", but these ones need human eyes anyway.

#### **Translate Toolkit 0.10.1**

Released on 28 December 2006

Release notes for this release are still to be recovered, feel free to do some way back searching and create a Pull Request for this release.

#### **Translate Toolkit 0.10**

Released on 29 August 2006

#### PO to XLIFF conversion

Conversion from PO to XLIFF is greatly improved in 0.10 and this was done according to the specification at http://xliff-tools.freedesktop.org/wiki/Projects/XliffPoGuide – please let us know if there are features lacking.

# pot2po can replace msgmerge

pot2po has undergone major changes which means that it now respects your header entries, can resurrect obsolete messages, does fuzzy matching using *Levenshtein distance* algorithm, will correctly match messages with KDE style comments and can use an external Translation Memory. You can now use pot2po instead of Gettext's msgmerge and it can also replace *pomigrate2*. You may still want to use pomigrate2 if there where file movements between versions as pot2po can still not do intelligent matching of PO and POT files, pomigrate2 has also been adapted so that it can use pot2po as it background merging tool.

```
pomigrate2 --use-compendium --pot2po <old> <pot> <new>
```

This will migrate file with a compendium built from PO files in <old> and will use pot2po as its conversion engine.

# .properties pretty formatting

When using templates for generating translated .properties files we will now preserve the formatting around the equal sign.

```
# Previously if the template had property = value
```

```
# We output property=translation
```

```
# We will now output property = translation
```

This change ensures that there is less noise when checking differences against the template file. However, there will be quite a bit of noise when you make your first properties commits with the new pretty layout. Our suggestion is that you make a single commit of properties files without changes of translations to gt the formatting correct.

# **Translate Toolkit 0.9.2**

Released on 11 August 2006

Release notes for this release are still to be recovered, feel free to do some way back searching and create a Pull Request for this release.

#### **Translate Toolkit 0.9.1**

Released on 17 July 2006

Release notes for this release are still to be recovered, feel free to do some way back searching and create a Pull Request for this release.

#### **Translate Toolkit 0.9**

Released on 15 June 2006

# Escaping - DTD files are no longer escaped

Previously each converter handled escaping, which made it a nightmare every time we identified an escaping related error or added a new format. Escaping has now been moved into the format classes as much as possible, the result being that formats exchange Python strings and manage their own escaping.

I doing this migration we revisited some of the format migration. We found that we were escaping elements in our output DTD files. DTD's should have no escaping i.e.  $\n$  is a literal  $\n$  followed by an n not a newline.

A result of this change is that older PO files will have different escaping to what po2moz will now expect. Probably resulting in bad output .dtd files.

We did not make this backward compatible as the fix is relatively simple and is one you would have done for any migration of your PO files.

1. Create a new set of POT files

```
moz2po -P mozilla pot
```

2. Migrate your old PO files

```
pomigrate2 old new pot
```

- 3. Fix all the fuzzy translations by editing your PO files
- 4. Use pofilter to check for escaping problems and fix them

```
pofilter -t escapes new new-check
```

5. Edit file in new-check in your PO editor

```
pomerge -t new -i new-check -o new-check
```

# Migration to base class

All filters are/have been migrate to a base class. This move is so that it is easier to add new format, interchange formats and to create converters. Thus xx2po and xx2xlf become easier to create. Also adding a new format should be as simple as working towards the API exposed in the base class. An unexpected side effect will be the Pootle should be able to work directly with any base class file (although that will not be the normal Pootle operation)

We have checks in place to ensure the current operation remains correct. However, nothing is perfect and unfortunately the only way to really expose all bugs is to release this software.

If you discover a bug please report it on Bugzilla or on the Pootle mailing list. If you have the skills please check on HEAD to see if it is not already fixed and if you regard it as critical discuss on the mailing list backporting the fix (note some fixes will not be backported because they may be too invasive for the stable branch). If you are a developer please write a test to expose the bug and a fix if possible.

# Duplicate Merging in PO files - merge now the default

We added the --duplicatestyle option to allow duplicate messages to be merged, commented or simply appear in the PO unmerged. Initially we used the msgid\_comments options as the default. This adds a KDE style comment to all affected messages which created a good balance allowing users to see duplicates in the PO file but still create a valid PO file.

'msgid\_comments' was the default for 0.8 (FIXME check), however it seemed to create more confusion then it solved. Thus we have reverted to using 'merge' as the default (this then completely mimics Gettext behaviour).

As Gettext will soon introduce the msgctxt attribute we may revert to using that to manage disambiguation messages instead of KDE comments. This we feel will put us back at a good balance of usefulness and usability. We will only release this when msgctxt version of the Gettext tools are released.

# .properties files no longer use escaped Unicode

The main use of the .properties converter class is to translate Mozilla files, although .properties files are actually a Java standard. The old Mozilla way, and still the Java way, of working with .properties files is to escape any Unicode characters using the \uNNNN convention. Mozilla now allows you to use Unicode in UTF-8 encoding for these files. Thus in 0.9 of the Toolkit we now output UTF-8 encoded properties files. Issue 193 tracks the status of this and we hope to add a feature to prop2po to restore the correct Java convention as an option.

#### **Translate Toolkit 0.8**

Released on 20 February 2006

Release notes for this release are still to be recovered, feel free to do some way back searching and create a Pull Request for this release.

#### **Translate Toolkit 0.7.1**

Released on 24 April 2004

Release notes for this release are still to be recovered, feel free to do some way back searching and create a Pull Request for this release.

# 3.2 History of the Translate Toolkit

This is a short history of the Translate Toolkit. In many ways written so that people who see problems in the toolkit can understand how it evolved and where it is going.

# 3.2.1 Conception

The toolkit was developed by David Fraser while working for Translate.org.za. Initially Translate.org.za had focussed on translating KDE into South Africa languages, this work was PO based. The next project was to translate Mozilla which used a combination of DTD and .properties files. The Mozilla project used a tool called Mozilla Translator, which mostly worked although it was not as feature rich as KBabel that was being used to manage PO translations. A decision was made to create a set of tools that could convert the DTD and .properties files into PO files. The advantage being that translators would not need to learn new tools, that existing translations could be leveraged and that the resultant files, being bilingual, would make it easier to upgrade and manage translations.

Thus was born what initially was called the mozpotools.

# 3.2.2 **Growth**

The first problem with the tools was that it was possible to break Mozilla translations. This was a combination of the fact that translators would often translate variables such as &browserName; and that the toolkit had developed a method of folding labels and accelerators into one PO field. These breakages where presented as broken XML. Thus was born pofilter which allowed us to check the translations for problems in variables and accelerators. pomerge its sister allowed us to merge the corrections back into the main. We also developed pocount which allowed us to for the first time get a real feel of the volume of work required in translating a PO file.

# 3.2.3 Expansion

Of course once you can convert the convoluted Mozilla translations then you realise you can do anything. A key addition was the converter for OpenOffice.org but also added where TMX, Qt .ts, txt and OpenOffice.org SXW files.

The key being that files are converted to PO to allow translations and use of the Gettext tools and existing PO files.

# 3.2.4 Pootle

Initially started as a separate project to allow online translation it was soon realised that the toolkit being file based gave all the infrastructure to allow Pootle to be a wrapper around the toolkit. So a file based, web translation tool was created.

# 3.2.5 WordForge project

In 2006 with funding from the Open Society Institute (OSI) and IDRC the toolkit was adapted to allow many core changes. The first being to introduce the concept of a base class from which PO and XLIFF storage formats are derived. This allowed tools to be adapted to allow output to XLIFF or PO files. The tools themselves where adapted to allow them to work with the core formats XLIFF and PO as well as all base class derived formats. Thus we can count XLIFF, PO, MO and other formats.

Additional contributions during this phase where the adaptation of Pootle to use XLIFF as well as PO. The creation of tools to manage translation memory files and glossary files.

The toolkit was also adapted to make dealing with encodings, plural forms, and escaping easier and more consistent throughout the code. Many but not all of the formats where converted to the base class.

As part of the WordForge project Pootling was created which in the same way that Pootle is a web-based wrapper around the toolkit so Pootling is a GUI wrapper around the toolkit.

# 3.2.6 ANLoc project

The African Network for Localisation provided the opportunity for further improvements to the project. We saw the first official releases of Virtaal and massive improvements to all the translation tools.

Format support improved a lot, with several bilingual file formats now support (Wordfast TM, Qt TS, etc.), and several monolingual file formats (PHP arrays, video subtitles, Mac OS X strings, etc.).

# 3.2.7 The Future

The toolkit continues to evolve with clean-up focused in various areas:

- Pulling features out of Pootle that should be in the Toolkit
- Cleaning up storage classes and converters to be XLIFF/PO interchangeable
- Cleaning up the converters to use only base class features and migrating code from the converters to the storage class
- · Adding storage classes as needed
- · Optimisation where needed

The toolkit continues to serve as the core for the command line tools and for Pootle. Key new features:

Process Management

# 3.3 License

The Translate Toolkit documentation is released under the GNU General Public License (GPL).

# CHAPTER 4

# **API** Reference

This part covers any function, class or method included within the Translate Toolkit that you can use to programatically build new localization tools.

# 4.1 API

The Translate Toolkit provides several modules for programmers to build their own tools.

# 4.1.1 Module overview

The following will give you an idea about what each module is capable of.

#### convert

Code to convert between different storage formats for localizations.

#### filters

Filters that can be used on translations...

# lang

Classes that represent languages and provides language-specific information.

All classes inherit from the parent class called common.

The type of data includes:

- · Language codes
- Language name

- · Plurals
- Punctuation transformation
- etc.

#### misc

Miscellaneous modules for translate - including modules for backward compatibility with pre-2.3 versions of Python

#### search

Services for searching and matching of text.

#### services

translate.services is part of the translate toolkit. It provides network services for interacting with the toolkit

# storage

Classes that represent various storage formats for localization.

#### tools

Code to perform various operations, mostly on po files.

# 4.1.2 Module list

All the modules included in the Translated Toolkit are listed here.

#### convert

Code to convert between different storage formats for localizations.

#### acesskey

functions used to manipulate access keys in strings

```
class translate.convert.accesskey.UnitMixer(labelsuffixes, accesskeysuffixes)
```

Helper to mix separately defined labels and accesskeys into one unit.

```
match_entities (index)
```

Populates mixedentities from the index.

```
mix_units (label_unit, accesskey_unit, target_unit)
```

Mix the given units into the given target\_unit if possible.

Might return None if no match is possible.

```
translate.convert.accesskey.combine(label, accesskey_marker=u'&')
```

Combine a label and and accesskey to form a label+accesskey string

We place an accesskey marker before the accesskey in the label and this creates a string with the two combined e.g. "File" + "F" = "&File"

The case of the accesskey is preferred unless no match is found, in which case the alternate case is used.

#### **Parameters**

- label (unicode) a label
- accesskey (unicode char) The accesskey

Return type unicode or None

Returns label+accesskey string or None if uncombineable

```
translate.convert.accesskey.extract(string, accesskey_marker=u'&')
```

Extract the label and accesskey from a label+accesskey string

The function will also try to ignore &entities; which would obviously not contain accesskeys.

#### **Parameters**

- string (Unicode) A string that might contain a label with accesskey marker
- accesskey\_marker (Char) The character that is used to prefix an access key

#### convert

Handles converting of files between formats (used by translate.convert tools).

ConvertOptionParser that can handle recursing into single archive files.

archiveformats maps extension to class. If the extension doesn't matter, it can be None.

If the extension is only valid for input/output/template, it can be given as (extension, filepurpose).

```
add_duplicates_option(default='msgctxt')
```

Adds an option to say what to do with duplicate strings.

```
add_fuzzy_option(default=False)
```

Adds an option to include / exclude fuzzy translations.

```
add_multifile_option (default='single')
```

Adds an option to say how to split the po/pot files.

```
add_option(Option)
add_option(opt_str, ..., kwarg=val, ...)
```

# add\_remove\_untranslated\_option(default=False)

Adds an option to remove key value from output if it is untranslated.

#### add\_threshold\_option(default=None)

Adds an option to output only stores where translation percentage exceeds the threshold.

4.1. API 233

# check\_values (values : Values, args : [string]) -> (values : Values, args : [string])

Check that the supplied option values and leftover arguments are valid. Returns the option values and leftover arguments (possibly adjusted, possibly completely new – whatever you like). Default implementation just returns the passed-in values; subclasses may override as desired.

# checkoutputsubdir(options, subdir)

Checks to see if subdir under options.output needs to be created, creates if neccessary.

#### define\_option(option)

Defines the given option, replacing an existing one of the same short name if neccessary...

#### destroy()

Declare that you are done with this OptionParser. This cleans up reference cycles so the OptionParser (and all objects referenced by it) can be garbage-collected promptly. After calling destroy(), the OptionParser is unusable.

# disable\_interspersed\_args()

Set parsing to stop on the first non-option. Use this if you have a command processor which runs another command that has options of its own and you want to make sure these options don't get confused.

#### enable\_interspersed\_args()

Set parsing to not stop on the first non-option, allowing interspersing switches with command arguments. This is the default behavior. See also disable\_interspersed\_args() and the class documentation description of the attribute allow\_interspersed\_args.

#### error (msg : string)

Print a usage message incorporating 'msg' to stderr and exit. If you override this in a subclass, it should not return – it should either exit or raise an exception.

# filterinputformats (options)

Filters input formats, processing relevant switches in options.

# filteroutputoptions (options)

Filters output options, processing relevant switches in options.

#### finalizetempoutputfile (options, outputfile, fulloutputpath)

Write the temp outputfile to its final destination.

#### format\_manpage()

returns a formatted manpage

# getarchiveclass (fileext, filepurpose, isdir=False)

Returns the archiveclass for the given fileext and filepurpose

# getformathelp (formats)

Make a nice help string for describing formats...

#### getfullinputpath (options, inputpath)

Gets the absolute path to an input file.

# getfulloutputpath (options, outputpath)

Gets the absolute path to an output file.

#### getfulltemplatepath (options, templatepath)

Gets the absolute path to a template file.

#### getoutputname (options, inputname, outputformat)

Gets an output filename based on the input filename.

# getoutputoptions (options, inputpath, templatepath)

Works out which output format and processor method to use...

# getpassthroughoptions(options) Get the options required to pass to the filtermethod... gettemplatename (options, inputname) Gets an output filename based on the input filename. getusageman (option) returns the usage string for the given option getusagestring(option) returns the usage string for the given option isarchive (fileoption, filepurpose='input') Returns whether the file option is an archive file. isexcluded (options, inputpath) Checks if this path has been excluded. isrecursive (fileoption, filepurpose='input') Checks if **fileoption** is a recursive file. isvalidinputname (inputname) Checks if this is a valid input filename. mkdir (parent, subdir) Makes a subdirectory (recursively if neccessary). openarchive (archivefilename, filepurpose, \*\*kwargs) Creates an archive object for the given file. openinputfile (options, fullinputpath) Opens the input file. openoutputfile (options, fulloutputpath) Opens the output file. opentemplatefile (options, fulltemplatepath) Opens the template file (if required). opentempoutputfile (options, fulloutputpath) Opens a temporary output file. parse\_args (args=None, values=None) Parses the command line options, handling implicit input/output args. potifyformat (fileformat) Converts a .po to a .pot where required. print\_help (file : file = stdout) Print an extended help message, listing all options and any help text provided with them, to 'file' (default stdout). print\_manpage (file=None) outputs a manpage for the program using the help information print\_usage (file : file = stdout)

4.1. API 235

Does nothing if self.usage is empty or not defined.

print\_version (file : file = stdout)

Print the usage message for the current program (self.usage) to 'file' (default stdout). Any occurrence of the string "%prog" in self.usage is replaced with the name of the current program (basename of sys.argv[0]).

Print the version message for this program (self.version) to 'file' (default stdout). As with print usage(),

any occurrence of "%prog" in self.version is replaced by the current program's name. Does nothing if self.version is empty or undefined.

# ${\tt processfile}$ (fileprocessor, options, fullinputpath, fulloutputpath, fulltemplatepath)

Run an invidividual conversion.

#### recursearchivefiles (options)

Recurse through archive files and convert files.

#### recurseinputfilelist (options)

Use a list of files, and find a common base directory for them.

#### recurseinputfiles (options)

Recurse through archive file / directories and return files to be converted.

#### recursiveprocess (options)

Recurse through directories and convert files.

#### run (argv=None)

Parses the command line options and runs the conversion.

#### set usage(usage=None)

sets the usage string - if usage not given, uses getusagestring for each option

#### seterrorleveloptions()

Sets the errorlevel options.

#### setformats (formats, usetemplates)

Sets the format options using the given format dictionary.

Parameters formats (Dictionary) - The dictionary keys should be:

- Single strings (or 1-tuples) containing an input format (if not *usetemplates*)
- Tuples containing an input format and template format (if *usetemplates*)
- Formats can be *None* to indicate what to do with standard input

The dictionary values should be tuples of outputformat (string) and processor method.

#### setmanpageoption()

creates a manpage option that allows the optionparser to generate a manpage

#### setpotoption()

Sets the -P/--pot option depending on input/output formats etc.

# setprogressoptions()

Sets the progress options.

#### settimestampoption()

Sets -S/--timestamp option.

# splitext (pathname)

Splits *pathname* into name and ext, and removes the extsep.

# Parameters pathname (string) - A file path

Returns root, ext

Return type tuple

# splitinputext(inputpath)

Splits an *inputpath* into name and extension.

# splittemplateext(templatepath)

Splits a *templatepath* into name and extension.

#### templateexists (options, templatepath)

Returns whether the given template exists...

# verifyoptions(options)

Verifies that the options are valid (required options are present, etc).

# warning (msg, options=None, exc\_info=None)

Print a warning message incorporating 'msg' to stderr and exit.

#### 

A specialized Option Parser for convertor tools...

# add\_duplicates\_option (default='msgctxt')

Adds an option to say what to do with duplicate strings.

#### add\_fuzzy\_option (default=False)

Adds an option to include / exclude fuzzy translations.

#### add\_multifile\_option (default='single')

Adds an option to say how to split the po/pot files.

```
add_option(Option)
```

```
add_option(opt_str, ..., kwarg=val, ...)
```

#### add\_remove\_untranslated\_option(default=False)

Adds an option to remove key value from output if it is untranslated.

#### add\_threshold\_option(default=None)

Adds an option to output only stores where translation percentage exceeds the threshold.

```
check_values (values : Values, args : [string])
    -> (values : Values, args : [string])
```

Check that the supplied option values and leftover arguments are valid. Returns the option values and leftover arguments (possibly adjusted, possibly completely new – whatever you like). Default implementation just returns the passed-in values; subclasses may override as desired.

# checkoutputsubdir(options, subdir)

Checks to see if subdir under options.output needs to be created, creates if neccessary.

# define\_option(option)

Defines the given option, replacing an existing one of the same short name if neccessary...

#### destroy()

Declare that you are done with this OptionParser. This cleans up reference cycles so the OptionParser (and all objects referenced by it) can be garbage-collected promptly. After calling destroy(), the OptionParser is unusable.

# disable\_interspersed\_args()

Set parsing to stop on the first non-option. Use this if you have a command processor which runs another command that has options of its own and you want to make sure these options don't get confused.

#### enable\_interspersed\_args()

Set parsing to not stop on the first non-option, allowing interspersing switches with command arguments. This is the default behavior. See also disable\_interspersed\_args() and the class documentation description of the attribute allow\_interspersed\_args.

# error (msg : string)

Print a usage message incorporating 'msg' to stderr and exit. If you override this in a subclass, it should not return – it should either exit or raise an exception.

4.1. API 237

#### filterinputformats (options)

Filters input formats, processing relevant switches in options.

# filteroutputoptions (options)

Filters output options, processing relevant switches in options.

#### finalizetempoutputfile (options, outputfile, fulloutputpath)

Write the temp outputfile to its final destination.

#### format manpage()

returns a formatted manpage

#### getformathelp(formats)

Make a nice help string for describing formats...

#### getfullinputpath (options, inputpath)

Gets the absolute path to an input file.

#### getfulloutputpath (options, outputpath)

Gets the absolute path to an output file.

#### getfulltemplatepath (options, templatepath)

Gets the absolute path to a template file.

# getoutputname (options, inputname, outputformat)

Gets an output filename based on the input filename.

#### getoutputoptions (options, inputpath, templatepath)

Works out which output format and processor method to use...

# getpassthroughoptions(options)

Get the options required to pass to the filtermethod...

# gettemplatename (options, inputname)

Gets an output filename based on the input filename.

# getusageman (option)

returns the usage string for the given option

# ${\tt getusagestring} \ (option)$

returns the usage string for the given option

# isexcluded(options, inputpath)

Checks if this path has been excluded.

# $\verb|isrecursive| (file option, file purpose = 'input')$

Checks if fileoption is a recursive file.

#### isvalidinputname (inputname)

Checks if this is a valid input filename.

# mkdir (parent, subdir)

Makes a subdirectory (recursively if neccessary).

# openinputfile (options, fullinputpath)

Opens the input file.

# ${\tt openoutputfile}\ (options, full output path)$

Opens the output file.

# opentemplatefile (options, fulltemplatepath)

Opens the template file (if required).

#### opentempoutputfile (options, fulloutputpath)

Opens a temporary output file.

#### parse\_args (args=None, values=None)

Parses the command line options, handling implicit input/output args.

#### potifyformat (fileformat)

Converts a .po to a .pot where required.

#### print help(file : file = stdout)

Print an extended help message, listing all options and any help text provided with them, to 'file' (default stdout).

#### print\_manpage (file=None)

outputs a manpage for the program using the help information

#### print\_usage (file : file = stdout)

Print the usage message for the current program (self.usage) to 'file' (default stdout). Any occurrence of the string "%prog" in self.usage is replaced with the name of the current program (basename of sys.argv[0]). Does nothing if self.usage is empty or not defined.

# print\_version (file : file = stdout)

Print the version message for this program (self.version) to 'file' (default stdout). As with print\_usage(), any occurrence of "%prog" in self.version is replaced by the current program's name. Does nothing if self.version is empty or undefined.

# processfile (fileprocessor, options, fullinputpath, fulloutputpath, fulltemplatepath)

Process an individual file.

#### recurseinputfilelist (options)

Use a list of files, and find a common base directory for them.

# recurseinputfiles (options)

Recurse through directories and return files to be processed.

# recursiveprocess (options)

Recurse through directories and process files.

#### **run** (argv=None)

Parses the command line options and runs the conversion.

#### set\_usage (usage=None)

sets the usage string - if usage not given, uses getusagestring for each option

# seterrorleveloptions()

Sets the errorlevel options.

#### setformats (formats, usetemplates)

Sets the format options using the given format dictionary.

# **Parameters** formats (Dictionary) – The dictionary keys should be:

- Single strings (or 1-tuples) containing an input format (if not *usetemplates*)
- Tuples containing an input format and template format (if usetemplates)
- Formats can be *None* to indicate what to do with standard input

The dictionary *values* should be tuples of outputformat (string) and processor method.

# setmanpageoption()

creates a manpage option that allows the optionparser to generate a manpage

4.1. API 239

```
setpotoption()
          Sets the -P/--pot option depending on input/output formats etc.
     setprogressoptions()
          Sets the progress options.
     settimestampoption()
          Sets -S/--timestamp option.
     splitext (pathname)
          Splits pathname into name and ext, and removes the extsep.
              Parameters pathname (string) - A file path
               Returns root, ext
              Return type tuple
     splitinputext(inputpath)
          Splits an inputpath into name and extension.
     splittemplateext(templatepath)
          Splits a templatepath into name and extension.
     templateexists (options, templatepath)
          Returns whether the given template exists...
     verifyoptions (options)
          Verifies that the options are valid (required options are present, etc).
     warning (msg, options=None, exc info=None)
          Print a warning message incorporating 'msg' to stderr and exit.
class translate.convert.convert.Replacer(searchstring, replacestring)
     An object that knows how to replace strings in files.
     doreplace (text)
          actually replace the text
     searchreplaceinput (inputfile, outputfile, templatefile, **kwargs)
          copies the input file to the output file, searching and replacing
     searchreplacetemplate (inputfile, outputfile, templatefile, **kwargs)
          Copies the template file to the output file, searching and replacing.
translate.convert.convert.copyinput (inputfile, outputfile, templatefile, **kwargs)
     Copies the input file to the output file.
translate.convert.convert.copytemplate(inputfile, outputfile, templatefile, **kwargs)
     Copies the template file to the output file.
translate.convert.convert.should_output_store(store, threshold)
     Check if the percent of translated source words more than or equal to the given threshold.
```

#### csv2po

Convert Comma-Separated Value (.csv) files to Gettext PO localization files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/csv2po.html for examples and usage instructions.

```
translate.convert.csv2po.convertcsv(inputfile, outputfile, templatefile, charset=None, colum-
                                                 norder=None, duplicatestyle='msgctxt')
     reads in inputfile using csvl10n, converts using csv2po, writes to outputfile
class translate.convert.csv2po.csv2po(templatepo=None,
                                                                          charset=None,
                                                                                            duplicat-
                                                   estyle='keep')
     a class that takes translations from a .csv file and puts them in a .po file
     convertstore(thecsvfile)
          converts a csvfile to a pofile, and returns it. uses templatepo if given at construction
     convertunit (csvunit)
          converts csv unit to po unit
     handlecsvunit (csvunit)
          handles reintegrating a csv unit into the .po file
     makeindex()
          makes indexes required for searching...
translate.convert.csv2po.replacestrings(source, *pairs)
     Use pairs of (original, replacement) to replace text found in source.
          Parameters
```

- source (String) String to on which pairs of strings are to be replaced
- \*pairs (One or more tuples of (original, replacement)) Strings to be matched and replaced

Returns String with \*pairs of strings replaced

# csv2tbx

Convert Comma-Separated Value (.csv) files to a TermBase eXchange (.tbx) glossary file

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/csv2tbx.html for examples and usage instructions

```
translate.convert.csv2tbx.convertcsv(inputfile, outputfile, templatefile, charset=None, columnorder=None)
reads in inputfile using csv110n, converts using csv2tbx, writes to outputfile

class translate.convert.csv2tbx.csv2tbx(charset=None)
a class that takes translations from a .csv file and puts them in a .tbx file

convertfile(csvfile)
```

converts a csvfile to a tbxfile, and returns it. uses templatepo if given at construction

# dtd2po

Convert a Mozilla .dtd UTF-8 localization format to a Gettext PO localization file.

Uses the po and dtd modules, and the dtd2po convertor class which is in this module You can convert back to .dtd using po2dtd.py.

```
translate.convert.dtd2po.convertdtd(inputfile, outputfile, templatefile, pot=False, duplicat-
estyle='msgctxt')
reads in inputfile and templatefile using dtd, converts using dtd2po, writes to outputfile
```

4.1. API 241

```
translate.convert.dtd2po.is_css_entity(entity)
     Says if the given entity is likely to contain CSS that should not be translated.
```

# factory

Factory methods to convert supported input files to supported translatable files.

```
exception translate.convert.factory.UnknownExtensionError(afile)
exception translate.convert.factory.UnsupportedConversionError(in_ext=None,
                                                                    out_ext=None,
                                                                    templ_ext=None)
```

# html2po

Convert HTML files to Gettext PO localization files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/html2po.html for examples and usage instructions.

```
translate.convert.html2po.converthtml(inputfile,
                                                                outputfile,
                                                                              templates,
                                                                                            includeun-
                                                    tagged=False, pot=False, duplicatestyle='msgctxt',
                                                    keepcomments=False)
     reads in stdin using fromfileclass, converts using convertorclass, writes to stdout
```

# ical2po

Convert iCalendar files to Gettext PO localization files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/ical2po.html for examples and usage instructions.

```
class translate.convert.ical2po.ical2po(input_file,
                                                                output_file,
                                                   blank_msgstr=False, duplicate_style='msgctxt')
     Convert one or two iCalendar files to a single PO file.
     SourceStoreClass
          alias of translate.storage.ical.icalfile
     TargetStoreClass
          alias of translate.storage.pypo.pofile
     TargetUnitClass
          alias of translate.storage.pypo.pounit
     convert_store()
          Convert a single source format file to a target format file.
     convert unit(unit)
          Convert a source format unit to a target format unit.
     merge_stores()
          Convert two source format files to a target format file.
```

template\_file=None,

run()

Run the converter.

```
translate.convert.ical2po.run\_converter(input\_file, output\_file, template\_file=None, pot=False, duplicatestyle='msgctxt') \\ Wrapper around converter.
```

# ini2po

Convert .ini files to Gettext PO localization files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/ini2po.html for examples and usage instructions.

```
class translate.convert.ini2po.ini2po(input_file,
                                                                output_file,
                                                                               template_file=None,
                                                 blank_msgstr=False,
                                                                          duplicate_style='msgctxt',
                                                 dialect='default')
     Convert one or two INI files to a single PO file.
     SourceStoreClass
          alias of translate.storage.ini.inifile
     TargetStoreClass
          alias of translate.storage.pypo.pofile
     TargetUnitClass
          alias of translate.storage.pypo.pounit
     convert store()
          Convert a single source format file to a target format file.
     convert_unit (unit)
          Convert a source format unit to a target format unit.
     merge stores()
          Convert two source format files to a target format file.
     run()
          Run the converter.
translate.convert.ini2po.run_converter(input_file,
                                                                output_file,
                                                                               template_file=None,
                                                   pot=False,
                                                                  duplicatestyle='msgctxt',
                                                   alect='default')
     Wrapper around converter.
```

# json2po

Convert JSON files to Gettext PO localization files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/json2po.html for examples and usage instructions.

```
translate.convert.json2po.convertjson(input_file, output_file, template_file, pot=False, dupli-
catestyle='msgctxt', dialect='default', filter=None)
Reads in input_file using json110n, converts using json2po, writes to output_file.

class translate.convert.json2po.json2po
Convert a JSON file to a PO file

convert_store(input_store, duplicatestyle='msgctxt')
Converts a JSON file to a PO file
```

4.1. API 243

# moz2po

Convert Mozilla .dtd and .properties files to Gettext PO localization files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/moz2po.html for examples and usage instructions.

# mozfunny2prop

Converts additional Mozilla files to properties files.

```
translate.convert.mozfunny2prop.inc2po (inputfile, outputfile, templatefile, encoding=None, pot=False, duplicatestyle='msgctxt') wraps prop2po but converts input/template files to properties first

translate.convert.mozfunny2prop.inc2prop (lines) convert a .inc file with #defines in it to a properties file

translate.convert.mozfunny2prop.it2po (inputfile, outputfile, templatefile, encoding='cp1252', pot=False, duplicatestyle='msgctxt') wraps prop2po but converts input/template files to properties first

translate.convert.mozfunny2prop.it2prop (lines, encoding='cp1252') convert a pseudo-properties .it file to a conventional properties file
```

#### mozlang2po

Convert Mozilla .lang files to Gettext PO localization files.

```
TargetUnitClass
```

```
alias of translate.storage.pypo.pounit
```

```
convert store()
```

Convert a single source format file to a target format file.

```
convert_unit (unit)
```

Convert a source format unit to a target format unit.

#### odf2xliff

Convert OpenDocument (ODF) files to XLIFF localization files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/odf2xliff.html for examples and usage instructions.

```
translate.convert.odf2xliff.convertodf(inputfile, outputfile, templates)
Convert an ODF package to XLIFF.
```

# oo2po

Convert an OpenOffice.org (SDF) localization file to Gettext PO localization files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/oo2po.html for examples and usage instructions.

```
translate.convert.oo2po.convertoo(inputfile, outputfile, templates, pot=False, source-language=None, targetlanguage=None, duplicat-estyle='msgid_comment', multifilestyle='single') reads in stdin using inputstore class, converts using convertorclass, writes to stdout

translate.convert.oo2po.verifyoptions(options) verifies the commandline options
```

# oo2xliff

Convert an OpenOffice.org (SDF) localization file to XLIFF localization files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/oo2po.html for examples and usage instructions.

```
translate.convert.oo2xliff.convertoo(inputfile, outputfile, templates, pot=False, source-language=None, targetlanguage=None, duplicat-estyle='msgctxt', multifilestyle='single')
reads in stdin using inputstore class, converts using convertorclass, writes to stdout
translate.convert.oo2xliff.verifyoptions(options)
verifies the commandline options
```

# php2po

Convert PHP localization files to Gettext PO localization files.

4.1. API 245

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/php2po.html for examples and usage instructions.

```
class translate.convert.php2po.php2po(input_file,
                                                               output_file,
                                                                              template_file=None,
                                                 blank_msgstr=False, duplicate_style='msgctxt')
     Convert one or two PHP files to a single PO file.
     SourceStoreClass
          alias of translate.storage.php.phpfile
     TargetStoreClass
          alias of translate.storage.pypo.pofile
     TargetUnitClass
          alias of translate.storage.pypo.pounit
     convert_store()
          Convert a single source format file to a target format file.
     convert_unit (unit)
          Convert a source format unit to a target format unit.
     merge_stores()
          Convert two source format files to a target format file.
     run()
          Run the converter.
                                                                              template_file=None,
translate.convert.php2po.run_converter(input_file,
                                                                output_file,
                                                  pot=False, duplicatestyle='msgctxt')
     Wrapper around converter.
```

# po2csv

Convert Gettext PO localization files to Comma-Separated Value (.csv) files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/csv2po.html for examples and usage instructions.

translate.convert.po2csv.convertcsv(inputfile, outputfile, templatefile, columnorder=None) reads in inputfile using po, converts using po2csv, writes to outputfile

# po2dtd

Converts a Gettext PO file to a UTF-8 encoded Mozilla .dtd file.

```
translate.convert.po2dtd.applytranslation (entity, dtdunit, inputunit, mixedentities) applies the translation for entity in the po unit to the dtd unit
```

```
class translate.convert.po2dtd.po2dtd(android=False, remove_untranslated=False) this is a convertor class that creates a new dtd file based on a po file without a template
```

**class** translate.convert.po2dtd.**redtd**(*dtdfile*, *android=False*, *remove\_untranslated=False*) this is a convertor class that creates a new dtd based on a template using translations in a po

# po2html

Convert Gettext PO localization files to HTML files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/html2po.html for examples and usage instructions.

```
translate.convert.po2html.converthtml (inputfile, outputfile, templatefile, includefuzzy=False, outputthreshold=None)
reads in stdin using fromfileclass, converts using convertorclass, writes to stdout

class translate.convert.po2html.po2html
po2html can take a po file and generate html. best to give it a template file otherwise will just concat msgstrs

mergestore (inputstore, templatetext, includefuzzy)
converts a file to .po format
```

# po2ical

Convert Gettext PO localization files to iCalendar files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/ical2po.html for examples and usage instructions.

translate.convert.po2ical.run\_converter(inputfile, outputfile, templatefile=None, include-

*fuzzy=False*, *outputthreshold=None*)

Wrapper around converter.

# po2ini

Convert Gettext PO localization files to .ini files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/ini2po.html for examples and usage instructions.

# SourceStoreClass

```
alias of translate.storage.pypo.pofile
```

4.1. API 247

```
TargetStoreClass
          alias of translate.storage.ini.inifile
     TargetUnitClass
          alias of translate.storage.ini.iniunit
     merge stores()
          Convert a source file to a target file using a template file.
          Source file is in source format, while target and template files use target format.
     run()
          Run the converter.
translate.convert.po2ini.run_converter(inputfile, outputfile, templatefile=None, include-
                                                  fuzzy=False,
                                                                  dialect='default',
                                                                                     outputthresh-
                                                   old=None)
     Wrapper around converter.
po2json
Convert Gettext PO localization files to JSON files.
See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/json2po.html for examples and us-
age instructions.
po2mozlang
Convert Gettext PO localization files to Mozilla .lang files.
class translate.convert.po2mozlang.po2lang(input_file, output_file, template_file=None,
                                                        include_fuzzy=False, output_threshold=None,
                                                        mark_active=True)
     Convert a PO file to a Mozilla .lang file.
     SourceStoreClass
          alias of translate.storage.pypo.pofile
     TargetStoreClass
          alias of translate.storage.mozilla_lang.LangStore
     TargetUnitClass
          alias of translate.storage.mozilla_lang.LangUnit
     convert store()
          Convert a single source format file to a target format file.
     convert_unit (unit)
          Convert a source format unit to a target format unit.
     run()
          Run the converter.
translate.convert.po2mozlang.run_converter(inputfile, outputfile, templatefile=None, in-
```

cludefuzzy=False, mark\_active=True, output-

threshold=None)

Wrapper around converter.

# po2moz

Convert Gettext PO localization files to Mozilla .dtd and .properties files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/moz2po.html for examples and usage instructions.

```
add_duplicates_option (default='msgctxt')
```

Adds an option to say what to do with duplicate strings.

```
add_fuzzy_option (default=False)
```

Adds an option to include / exclude fuzzy translations.

```
add_multifile_option (default='single')
```

Adds an option to say how to split the po/pot files.

```
add_option (Option)
add_option(opt_str, ..., kwarg=val, ...)
```

```
add_remove_untranslated_option(default=False)
```

Adds an option to remove key value from output if it is untranslated.

```
add_threshold_option(default=None)
```

Adds an option to output only stores where translation percentage exceeds the threshold.

```
check_values (values : Values, args : [string])
    -> (values : Values, args : [string])
```

Check that the supplied option values and leftover arguments are valid. Returns the option values and leftover arguments (possibly adjusted, possibly completely new – whatever you like). Default implementation just returns the passed-in values; subclasses may override as desired.

```
checkoutputsubdir(options, subdir)
```

Checks to see if subdir under options.output needs to be created, creates if neccessary.

```
define_option(option)
```

Defines the given option, replacing an existing one of the same short name if neccessary...

# destroy()

Declare that you are done with this OptionParser. This cleans up reference cycles so the OptionParser (and all objects referenced by it) can be garbage-collected promptly. After calling destroy(), the OptionParser is unusable.

### disable interspersed args()

Set parsing to stop on the first non-option. Use this if you have a command processor which runs another command that has options of its own and you want to make sure these options don't get confused.

# enable\_interspersed\_args()

Set parsing to not stop on the first non-option, allowing interspersing switches with command arguments. This is the default behavior. See also disable\_interspersed\_args() and the class documentation description of the attribute allow\_interspersed\_args.

```
error (msg : string)
```

Print a usage message incorporating 'msg' to stderr and exit. If you override this in a subclass, it should not return – it should either exit or raise an exception.

# filterinputformats (options)

Filters input formats, processing relevant switches in options.

```
filteroutputoptions (options)
     Filters output options, processing relevant switches in options.
finalizetempoutputfile (options, outputfile, fulloutputpath)
     Write the temp outputfile to its final destination.
format manpage()
     returns a formatted manpage
getformathelp (formats)
     Make a nice help string for describing formats...
getfullinputpath (options, inputpath)
     Gets the absolute path to an input file.
getfulloutputpath (options, outputpath)
     Gets the absolute path to an output file.
getfulltemplatepath (options, templatepath)
     Gets the absolute path to a template file.
getoutputname (options, inputname, outputformat)
     Gets an output filename based on the input filename.
getoutputoptions (options, inputpath, templatepath)
     Works out which output format and processor method to use...
getpassthroughoptions (options)
     Get the options required to pass to the filtermethod...
gettemplatename (options, inputname)
     Gets an output filename based on the input filename.
getusageman (option)
     returns the usage string for the given option
getusagestring(option)
     returns the usage string for the given option
isexcluded(options, inputpath)
     Checks if this path has been excluded.
isrecursive (fileoption, filepurpose='input')
     Checks if fileoption is a recursive file.
isvalidinputname (inputname)
     Checks if this is a valid input filename.
mkdir (parent, subdir)
     Makes a subdirectory (recursively if neccessary).
openinputfile (options, fullinputpath)
     Opens the input file.
openoutputfile (options, fulloutputpath)
     Opens the output file.
opentemplatefile (options, fulltemplatepath)
     Opens the template file (if required).
opentempoutputfile (options, fulloutputpath)
     Opens a temporary output file.
```

#### parse args (args=None, values=None)

Parses the command line options, handling implicit input/output args.

## potifyformat (fileformat)

Converts a .po to a .pot where required.

#### print\_help (file : file = stdout)

Print an extended help message, listing all options and any help text provided with them, to 'file' (default stdout).

## print\_manpage (file=None)

outputs a manpage for the program using the help information

# print\_usage (file : file = stdout)

Print the usage message for the current program (self.usage) to 'file' (default stdout). Any occurrence of the string "%prog" in self.usage is replaced with the name of the current program (basename of sys.argv[0]). Does nothing if self.usage is empty or not defined.

# print\_version (file : file = stdout)

Print the version message for this program (self.version) to 'file' (default stdout). As with print\_usage(), any occurrence of "%prog" in self.version is replaced by the current program's name. Does nothing if self.version is empty or undefined.

# $\verb|processfile| (file processor, options, full input path, full output path, full template path)|$

Process an individual file.

#### recurseinputfilelist (options)

Use a list of files, and find a common base directory for them.

#### recurseinputfiles (options)

Recurse through directories and return files to be processed.

## recursiveprocess (options)

recurse through directories and convert files

# run (argv=None)

Parses the command line options and runs the conversion.

## set\_usage (usage=None)

sets the usage string - if usage not given, uses getusagestring for each option

### seterrorleveloptions()

Sets the errorlevel options.

# $\verb"setformats" (formats, use templates")$

Sets the format options using the given format dictionary.

#### **Parameters formats** (Dictionary) – The dictionary keys should be:

- Single strings (or 1-tuples) containing an input format (if not *usetemplates*)
- Tuples containing an input format and template format (if usetemplates)
- Formats can be None to indicate what to do with standard input

The dictionary values should be tuples of outputformat (string) and processor method.

#### setmanpageoption()

creates a manpage option that allows the optionparser to generate a manpage

## setpotoption()

Sets the -P/--pot option depending on input/output formats etc.

```
setprogressoptions()
     Sets the progress options.
settimestampoption()
     Sets -S/--timestamp option.
splitext (pathname)
     Splits pathname into name and ext, and removes the extsep.
         Parameters pathname (string) – A file path
         Returns root, ext
         Return type tuple
splitinputext (inputpath)
     splits a inputpath into name and extension
     Special adaptation to handle po2moz case where extensions are e.g. properties.po
splittemplateext (templatepath)
     Splits a templatepath into name and extension.
templateexists (options, templatepath)
     Returns whether the given template exists...
verifyoptions(options)
     Verifies that the options are valid (required options are present, etc).
warning (msg, options=None, exc_info=None)
     Print a warning message incorporating 'msg' to stderr and exit.
```

# po2oo

Convert Gettext PO localization files to an OpenOffice.org (SDF) localization file.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/oo2po.html for examples and usage instructions.

# po2php

Convert Gettext PO localization files to PHP localization files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/php2po.html for examples and usage instructions.

## po2prop

Convert Gettext PO localization files to Java/Mozilla .properties files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/prop2po.html for examples and usage instructions.

```
translate.convert.po2prop.applytranslation (key, propunit, inunit, mixedkeys) applies the translation for key in the po unit to the prop unit
```

```
translate.convert.po2prop.convertmozillaprop(inputfile, outputfile, templatefile, include-
fuzzy=False, remove_untranslated=False,
outputthreshold=None)
```

Mozilla specific convertor function

```
translate.convert.po2prop.convertstrings (inputfile, outputfile, templatefile, personality='strings', includefuzzy=False, encoding=None, outputthreshold=None, remove_untranslated=False)
```

.strings specific convertor function

# po2rc

Convert Gettext PO localization files back to Windows Resource (.rc) files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/rc2po.html for examples and usage instructions.

# po2resx

Convert Gettext PO localisation files to .Net Resource (.resx) files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/resx2po.html for examples and usage instructions.

# po2sub

Convert Gettext PO localization files to subtitle files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/sub2po.html for examples and usage instructions.

## po2symb

Convert Gettext PO localization files to Symbian translation files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/symb2po.html for examples and usage instructions.

# po2tiki

Convert Gettext PO files to TikiWiki's language.php files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/tiki2po.html for examples and usage instructions.

class translate.convert.po2tiki.po2tiki(input\_file, output\_file, template\_file=None)
 Convert a PO file and a template TikiWiki file to a TikiWiki file.

#### SourceStoreClass

```
alias of translate.storage.pypo.pofile
```

## TargetStoreClass

```
alias of translate.storage.tiki.TikiStore
```

```
TargetUnitClass
    alias of translate.storage.tiki.TikiUnit

convert_store()
    Convert a single source format file to a target format file.

convert_unit(unit)
    Convert a source format unit to a target format unit.

run()
    Run the converter.

translate.convert.po2tiki.run_converter(inputfile, outputfile, template=None)
    Wrapper around converter.
```

# po2tmx

Convert Gettext PO localization files to a TMX (Translation Memory eXchange) file.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/po2tmx.html for examples and usage instructions.

```
{\bf class} \ {\bf translate.convert.po2tmx.TmxOptionParser} \ (formats, \quad use templates = False, \quad use templates = False,
```

```
add_duplicates_option(default='msgctxt')
```

Adds an option to say what to do with duplicate strings.

```
add_fuzzy_option (default=False)
```

Adds an option to include / exclude fuzzy translations.

```
add_multifile_option (default='single')
```

Adds an option to say how to split the po/pot files.

```
add_option(Option)
  add_option(opt_str,..., kwarg=val,...)
```

#### add\_remove\_untranslated\_option(default=False)

Adds an option to remove key value from output if it is untranslated.

```
add_threshold_option(default=None)
```

Adds an option to output only stores where translation percentage exceeds the threshold.

```
check_values (values : Values, args : [string])
    -> (values : Values, args : [string])
```

Check that the supplied option values and leftover arguments are valid. Returns the option values and leftover arguments (possibly adjusted, possibly completely new – whatever you like). Default implementation just returns the passed-in values; subclasses may override as desired.

```
checkoutputsubdir(options, subdir)
```

Checks to see if subdir under options . output needs to be created, creates if neccessary.

```
define_option(option)
```

Defines the given option, replacing an existing one of the same short name if neccessary...

#### destroy()

Declare that you are done with this OptionParser. This cleans up reference cycles so the OptionParser (and all objects referenced by it) can be garbage-collected promptly. After calling destroy(), the OptionParser is unusable.

### disable\_interspersed\_args()

Set parsing to stop on the first non-option. Use this if you have a command processor which runs another command that has options of its own and you want to make sure these options don't get confused.

#### enable\_interspersed\_args()

Set parsing to not stop on the first non-option, allowing interspersing switches with command arguments. This is the default behavior. See also disable\_interspersed\_args() and the class documentation description of the attribute allow\_interspersed\_args.

# error (msg : string)

Print a usage message incorporating 'msg' to stderr and exit. If you override this in a subclass, it should not return – it should either exit or raise an exception.

### filterinputformats (options)

Filters input formats, processing relevant switches in options.

## filteroutputoptions (options)

Filters output options, processing relevant switches in options.

# finalizetempoutputfile (options, outputfile, fulloutputpath)

Write the temp outputfile to its final destination.

#### format\_manpage()

returns a formatted manpage

# getarchiveclass (fileext, filepurpose, isdir=False)

Returns the archiveclass for the given fileext and filepurpose

### getformathelp (formats)

Make a nice help string for describing formats...

### getfullinputpath (options, inputpath)

Gets the absolute path to an input file.

# getfulloutputpath (options, outputpath)

Gets the absolute path to an output file.

# getfulltemplatepath (options, templatepath)

Gets the absolute path to a template file.

#### getoutputname (options, inputname, outputformat)

Gets an output filename based on the input filename.

# getoutputoptions (options, inputpath, templatepath)

Works out which output format and processor method to use...

### getpassthroughoptions(options)

Get the options required to pass to the filtermethod...

#### gettemplatename (options, inputname)

Gets an output filename based on the input filename.

# getusageman (option)

returns the usage string for the given option

#### getusagestring(option)

returns the usage string for the given option

### isarchive (fileoption, filepurpose='input')

Returns whether the file option is an archive file.

### isexcluded(options, inputpath)

Checks if this path has been excluded.

# isrecursive (fileoption, filepurpose='input')

Checks if **fileoption** is a recursive file.

## isvalidinputname (inputname)

Checks if this is a valid input filename.

#### mkdir (parent, subdir)

Makes a subdirectory (recursively if neccessary).

#### openarchive (archivefilename, filepurpose, \*\*kwargs)

Creates an archive object for the given file.

## openinputfile (options, fullinputpath)

Opens the input file.

# openoutputfile (options, fulloutputpath)

Opens the output file.

### opentemplatefile (options, fulltemplatepath)

Opens the template file (if required).

### opentempoutputfile (options, fulloutputpath)

Opens a temporary output file.

# parse\_args (args=None, values=None)

Parses the command line options, handling implicit input/output args.

#### potifyformat (fileformat)

Converts a .po to a .pot where required.

## print\_help (file : file = stdout)

Print an extended help message, listing all options and any help text provided with them, to 'file' (default stdout).

# print\_manpage (file=None)

outputs a manpage for the program using the help information

# print\_usage (file : file = stdout)

Print the usage message for the current program (self.usage) to 'file' (default stdout). Any occurrence of the string "%prog" in self.usage is replaced with the name of the current program (basename of sys.argv[0]). Does nothing if self.usage is empty or not defined.

## print\_version (file : file = stdout)

Print the version message for this program (self.version) to 'file' (default stdout). As with print\_usage(), any occurrence of "%prog" in self.version is replaced by the current program's name. Does nothing if self.version is empty or undefined.

### processfile (fileprocessor, options, fullinputpath, fulloutputpath, fulltemplatepath)

Run an invidividual conversion.

# recursearchivefiles (options)

Recurse through archive files and convert files.

# recurseinputfilelist(options)

Use a list of files, and find a common base directory for them.

#### recurseinputfiles (options)

Recurse through archive file / directories and return files to be converted.

## recursiveprocess (options)

Recurse through directories and convert files.

```
run (argv=None)
           Parses the command line options and runs the conversion.
     set_usage (usage=None)
           sets the usage string - if usage not given, uses getusagestring for each option
     seterrorleveloptions()
           Sets the errorlevel options.
     setformats (formats, usetemplates)
           Sets the format options using the given format dictionary.
               Parameters formats (Dictionary) – The dictionary keys should be:
                   • Single strings (or 1-tuples) containing an input format (if not usetemplates)
                   • Tuples containing an input format and template format (if usetemplates)
                   • Formats can be None to indicate what to do with standard input
                   The dictionary values should be tuples of outputformat (string) and processor method.
     setmanpageoption()
           creates a manpage option that allows the optionparser to generate a manpage
     setpotoption()
           Sets the -P/--pot option depending on input/output formats etc.
     setprogressoptions()
           Sets the progress options.
     settimestampoption()
           Sets -S/--timestamp option.
     splitext (pathname)
           Splits pathname into name and ext, and removes the extsep.
               Parameters pathname (string) – A file path
               Returns root, ext
               Return type tuple
     splitinputext(inputpath)
           Splits an inputpath into name and extension.
     splittemplateext (templatepath)
           Splits a templatepath into name and extension.
     templateexists (options, templatepath)
           Returns whether the given template exists...
     verifyoptions (options)
           Verifies that the options are valid (required options are present, etc).
     warning (msg, options=None, exc_info=None)
           Print a warning message incorporating 'msg' to stderr and exit.
translate.convert.po2tmx.convertpo(inputfile, outputfile, templatefile, sourcelanguage='en',
                                                targetlanguage=None, comment=None)
     reads in stdin using fromfileclass, converts using convertorclass, writes to stdout
```

# po2ts

Convert Gettext PO localization files to Qt Linguist (.ts) files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/ts2po.html for examples and usage instructions.

```
translate.convert.po2ts.convertpo (inputfile, outputfile, templatefile, context) reads in stdin using fromfileclass, converts using convertorclass, writes to stdout
```

## po2txt

Convert Gettext PO localization files to plain text (.txt) files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/txt2po.html for examples and usage instructions.

```
class translate.convert.po2txt.po2txt(input_file, output_file, template_file=None,
                                                                                output_threshold=None,
                                                     clude_fuzzy=False,
                                                     encoding='utf-8', wrap=None)
     po2txt can take a po file and generate txt.
     best to give it a template file otherwise will just concat msgstrs
     convert_store()
           Convert a source file to a target file.
     merge_stores()
           Convert a source file to a target file using a template file.
           Source file is in source format, while target and template files use target format.
     run()
           Run the converter.
     wrapmessage (message)
           rewraps text as required
translate.convert.po2txt.run_converter(inputfile, outputfile, templatefile=None, wrap=None,
                                                      includefuzzy=False, encoding='utf-8', outputthresh-
                                                      old=None)
     Wrapper around converter.
```

### po2web2py

Convert GNU/gettext PO files to web2py translation dictionaries (.py).

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/web2py2po.html for examples and usage instructions.

## po2wordfast

Convert Gettext PO localization files to a Wordfast translation memory file.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/po2wordfast.html for examples and usage instructions.

#### add\_duplicates\_option(default='msgctxt')

Adds an option to say what to do with duplicate strings.

## add\_fuzzy\_option (default=False)

Adds an option to include / exclude fuzzy translations.

#### add\_multifile\_option (default='single')

Adds an option to say how to split the po/pot files.

### add\_option (Option)

add\_option(opt\_str, ..., kwarg=val, ...)

#### add\_remove\_untranslated\_option(default=False)

Adds an option to remove key value from output if it is untranslated.

### add\_threshold\_option(default=None)

Adds an option to output only stores where translation percentage exceeds the threshold.

# check\_values (values : Values, args : [string])

-> (values : Values, args : [string])

Check that the supplied option values and leftover arguments are valid. Returns the option values and leftover arguments (possibly adjusted, possibly completely new – whatever you like). Default implementation just returns the passed-in values; subclasses may override as desired.

#### checkoutputsubdir(options, subdir)

Checks to see if subdir under options . output needs to be created, creates if neccessary.

## define\_option(option)

Defines the given option, replacing an existing one of the same short name if neccessary...

## destroy()

Declare that you are done with this OptionParser. This cleans up reference cycles so the OptionParser (and all objects referenced by it) can be garbage-collected promptly. After calling destroy(), the OptionParser is unusable.

# disable\_interspersed\_args()

Set parsing to stop on the first non-option. Use this if you have a command processor which runs another command that has options of its own and you want to make sure these options don't get confused.

#### enable interspersed args()

Set parsing to not stop on the first non-option, allowing interspersing switches with command arguments. This is the default behavior. See also disable\_interspersed\_args() and the class documentation description of the attribute allow\_interspersed\_args.

# error (msg : string)

Print a usage message incorporating 'msg' to stderr and exit. If you override this in a subclass, it should not return – it should either exit or raise an exception.

#### filterinputformats (options)

Filters input formats, processing relevant switches in options.

#### filteroutputoptions (options)

Filters output options, processing relevant switches in options.

# finalizetempoutputfile (options, outputfile, fulloutputpath)

Write the temp outputfile to its final destination.

```
format manpage()
     returns a formatted manpage
getarchiveclass (fileext, filepurpose, isdir=False)
     Returns the archiveclass for the given fileext and filepurpose
getformathelp (formats)
     Make a nice help string for describing formats...
getfullinputpath (options, inputpath)
     Gets the absolute path to an input file.
getfulloutputpath (options, outputpath)
     Gets the absolute path to an output file.
getfulltemplatepath (options, templatepath)
     Gets the absolute path to a template file.
getoutputname (options, inputname, outputformat)
     Gets an output filename based on the input filename.
getoutputoptions (options, inputpath, templatepath)
     Works out which output format and processor method to use...
getpassthroughoptions(options)
     Get the options required to pass to the filtermethod...
gettemplatename (options, inputname)
     Gets an output filename based on the input filename.
getusageman (option)
     returns the usage string for the given option
getusagestring(option)
     returns the usage string for the given option
isarchive (fileoption, filepurpose='input')
     Returns whether the file option is an archive file.
isexcluded(options, inputpath)
     Checks if this path has been excluded.
isrecursive (fileoption, filepurpose='input')
     Checks if fileoption is a recursive file.
isvalidinputname (inputname)
     Checks if this is a valid input filename.
mkdir (parent, subdir)
     Makes a subdirectory (recursively if neccessary).
openarchive (archivefilename, filepurpose, **kwargs)
     Creates an archive object for the given file.
openinputfile (options, fullinputpath)
     Opens the input file.
openoutputfile (options, fulloutputpath)
     Opens the output file.
opentemplatefile (options, fulltemplatepath)
     Opens the template file (if required).
```

#### opentempoutputfile (options, fulloutputpath)

Opens a temporary output file.

## parse\_args (args=None, values=None)

Parses the command line options, handling implicit input/output args.

# potifyformat (fileformat)

Converts a .po to a .pot where required.

#### print help(file: file = stdout)

Print an extended help message, listing all options and any help text provided with them, to 'file' (default stdout).

#### print\_manpage (file=None)

outputs a manpage for the program using the help information

### print\_usage (file : file = stdout)

Print the usage message for the current program (self.usage) to 'file' (default stdout). Any occurrence of the string "%prog" in self.usage is replaced with the name of the current program (basename of sys.argv[0]). Does nothing if self.usage is empty or not defined.

## print\_version (file : file = stdout)

Print the version message for this program (self.version) to 'file' (default stdout). As with print\_usage(), any occurrence of "%prog" in self.version is replaced by the current program's name. Does nothing if self.version is empty or undefined.

# processfile (fileprocessor, options, fullinputpath, fulloutputpath, fulltemplatepath)

Run an invidividual conversion.

# recursearchivefiles (options)

Recurse through archive files and convert files.

## recurseinputfilelist (options)

Use a list of files, and find a common base directory for them.

## recurseinputfiles (options)

Recurse through archive file / directories and return files to be converted.

#### recursiveprocess (options)

Recurse through directories and convert files.

### run (argv=None)

Parses the command line options and runs the conversion.

# set\_usage (usage=None)

sets the usage string - if usage not given, uses getusagestring for each option

#### seterrorleveloptions()

Sets the errorlevel options.

# setformats (formats, usetemplates)

Sets the format options using the given format dictionary.

# **Parameters** formats (Dictionary) – The dictionary keys should be:

- Single strings (or 1-tuples) containing an input format (if not *usetemplates*)
- Tuples containing an input format and template format (if *usetemplates*)
- Formats can be *None* to indicate what to do with standard input

The dictionary *values* should be tuples of outputformat (string) and processor method.

```
setmanpageoption()
          creates a manpage option that allows the optionparser to generate a manpage
     setpotoption()
          Sets the -P/--pot option depending on input/output formats etc.
     setprogressoptions()
          Sets the progress options.
     settimestampoption()
          Sets -S/--timestamp option.
     splitext (pathname)
          Splits pathname into name and ext, and removes the extsep.
              Parameters pathname (string) – A file path
              Returns root, ext
              Return type tuple
     splitinputext(inputpath)
          Splits an inputpath into name and extension.
     splittemplateext (templatepath)
          Splits a templatepath into name and extension.
     templateexists (options, templatepath)
          Returns whether the given template exists...
     verifyoptions (options)
          Verifies that the options are valid (required options are present, etc).
     warning (msg, options=None, exc_info=None)
          Print a warning message incorporating 'msg' to stderr and exit.
translate.convert.po2wordfast.convertpo(inputfile, outputfile,
                                                                                           sourcelan-
                                                                             templatefile,
                                                      guage='en', targetlanguage=None)
     reads in stdin using fromfileclass, converts using convertorclass, writes to stdout
```

# po2xliff

Convert Gettext PO localization files to XLIFF localization files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/xliff2po.html for examples and usage instructions.

```
translate.convert.po2xliff.convertpo (inputfile, outputfile, templatefile) reads in stdin using fromfileclass, converts using convertorclass, writes to stdout
```

# po2yaml

Convert Gettext PO localization files to YAML files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/yaml2po.html for examples and usage instructions.

```
SourceStoreClass
          alias of translate.storage.pypo.pofile
     TargetStoreClass
          alias of translate.storage.yaml.YAMLFile
     TargetUnitClass
          alias of translate.storage.yaml.YAMLUnit
     convert unit(unit)
          Convert a source format unit to a target format unit.
     merge_stores()
          Convert a source file to a target file using a template file.
          Source file is in source format, while target and template files use target format.
     run()
          Run the converter.
translate.convert.po2yaml.run_converter(inputfile, outputfile, templatefile=None, include-
                                                   fuzzy=False, outputthreshold=None)
     Wrapper around converter.
```

# pot2po

Convert template files (like .pot or template .xlf files) to translation files, preserving existing translations.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/pot2po.html for examples and usage instructions.

```
translate.convert.pot2po.\textbf{convert\_stores} (input\_store, template\_store, temp\_store=None, tm=None, min\_similarity=75, fuzzymatch-ing=True, **kwargs)
```

Actual conversion function, works on stores not files, returns a properly initialized pretranslated output store, with structure based on input\_store, metadata based on template\_store, migrates old translations from template\_store and pretranslating from TM.

```
translate.convert.pot2po.convertpot(input_file, output_file, template_file, tm=None, min_similarity=75, fuzzymatching=True, classes=None, classes_str=None, **kwargs)
```

Main conversion function.

# prop2mozfunny

Converts properties files to additional Mozilla format files.

```
translate.convert.prop2mozfunny.po2inc (inputfile, outputfile, templatefile, encoding=None, includefuzzy=False, remove_untranslated=False, outputthreshold=None)
wraps po2prop but converts outputfile to properties first

translate.convert.prop2mozfunny.po2ini (inputfile, outputfile, templatefile, encoding='UTF-8', includefuzzy=False, remove_untranslated=False, outputthreshold=None)
wraps po2prop but converts outputfile to properties first using UTF-8 encoding
```

```
translate.convert.prop2mozfunny.po2it (inputfile, outputfile, templatefile, encoding='cp1252',
                                                     includefuzzy=False, remove untranslated=False, out-
                                                     putthreshold=None)
     wraps po2prop but converts outputfile to properties first
translate.convert.prop2mozfunny.prop2inc(pf)
     convert a properties file back to a .inc file with #defines in it
translate.convert.prop2mozfunny.prop2it(pf)
     convert a properties file back to a pseudo-properties .it file
prop2po
Convert Java/Mozilla .properties files to Gettext PO localization files.
See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/prop2po.html for examples and us-
age instructions.
translate.convert.prop2po.convertmozillaprop (inputfile, outputfile, templatefile, pot=False,
                                                               duplicatestyle='msgctxt')
     Mozilla specific convertor function
translate.convert.prop2po.convertprop(inputfile, outputfile, templatefile, personality='java',
                                                                    duplicatestyle='msgctxt',
                                                                                                 encod-
                                                     pot=False,
                                                     ing=None)
     reads in inputfile using properties, converts using prop2po, writes to outputfile
translate.convert.prop2po.convertstrings(inputfile,
                                                                      outputfile,
                                                                                    templatefile,
                                                                                                    per-
                                                         sonality='strings',
                                                                               pot=False,
                                                                                               duplicat-
                                                         estyle='msgctxt', encoding=None)
     .strings specific convertor function
class translate.convert.prop2po.prop2po(personality='java', blankmsgstr=False, duplicat-
                                                        estyle='msgctxt')
     convert a .properties file to a .po file for handling the translation.
     convertpropunit (store, unit, commenttype, mixbucket='properties')
           Converts a unit from store to a po unit, keeping track of mixed names along the way.
           mixbucket can be specified to indicate if the given unit is part of the template or the translated file.
     convertstore (thepropfile)
           converts a .properties file to a .po file...
     convertunit (propunit, commenttype)
           Converts a properties unit to a po unit. Returns None if empty or not for translation.
     fold_gaia_plurals(postore)
           Fold the multiple plural units of a gaia file into a gettext plural.
     mergestore(origpropfile, translated propfile)
```

# rc2po

Convert Windows RC files to Gettext PO localization files.

converts two .properties files to a .po file. . .

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/rc2po.html for examples and usage instructions.

```
translate.convert.rc2po.convertrc(input_file, output_file, template_file, pot=False, duplicatestyle='msgctxt', charset=None, lang=None, sub-lang=None)
reads in input_file using rc, converts using rc2po, writes to output_file

class translate.convert.rc2po.rc2po
Convert a.rc file to a.po file for handling the translation.

convert_store(input_store, duplicatestyle='msgctxt')
converts a.rc file to a.po file...

convert_unit(input_unit, commenttype)
Converts a.rc unit to a.po unit. Returns None if empty or not for translation.

merge_store(template_store, input_store, blankmsgstr=False, duplicatestyle='msgctxt')
converts two.rc files to a.po file...
```

### resx2po

Convert .Net Resource (.resx) to Gettext PO localisation files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/resx2po.html for examples and usage instructions.

```
class translate.convert.resx2po.resx2po
    Convert a RESX file to a PO file for handling translation
    convert_store (input_store, duplicatestyle='msgctxt')
        Converts a RESX file to a PO file
    convert_unit (input_unit, commenttype)
        Converts a RESX unit to a PO unit @return: None if empty or not for translation
    merge_store (template_store, input_store, blankmsgstr=False, duplicatestyle='msgctxt')
        Converts two RESX files to a PO file
```

# sub2po

Convert subtitle files to Gettext PO localization files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/sub2po.html for examples and usage instructions.

```
translate.convert.sub2po.convert_store (input_store, duplicatestyle='msgctxt')
converts a subtitle file to a .po file...

translate.convert.sub2po.convert_unit (input_unit, commenttype)
Converts a subtitle unit to a .po unit. Returns None if empty or not for translation.

translate.convert.sub2po.convertsub (input_file, output_file, template_file=None, pot=False, duplicatestyle='msgctxt')
Reads in input_file using translate.subtitles, converts using sub2po, writes to output_file.

translate.convert.sub2po.merge_store (template_store, input_store, blankmsgstr=False, duplicatestyle='msgctxt')
converts two subtitle files to a .po file...
```

# symb2po

Convert Symbian localisation files to Gettext PO localization files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/symb2po.html for examples and usage instructions.

#### tiki2po

Convert TikiWiki's language.php files to Gettext PO localization files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/tiki2po.html for examples and usage instructions.

```
translate.convert.tiki2po.run_converter(input_file, output_file, template_file=None, in-
                                                  cludeunused=False)
     Wrapper around converter.
class translate.convert.tiki2po.tiki2po(input_file, output_file, template_file=None, in-
                                                  clude_unused=False)
     Convert one or two TikiWiki's language.php files to a single PO file.
     SourceStoreClass
          alias of translate.storage.tiki.TikiStore
     TargetStoreClass
          alias of translate.storage.pypo.pofile
     TargetUnitClass
          alias of translate.storage.pypo.pounit
     convert store()
         Convert a single source format file to a target format file.
     convert_unit (unit)
          Convert a source format unit to a target format unit.
     run()
          Run the converter.
```

# ts2po

Convert Qt Linguist (.ts) files to Gettext PO localization files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/ts2po.html for examples and usage instructions.

```
translate.convert.ts2po.\textbf{convertts} (inputfile, outputfile, templates, pot=False, duplicatestyle='msgctxt') \\ reads in stdin using fromfileclass, converts using convertorclass, writes to stdout
```

## txt2po

Convert plain text (.txt) files to Gettext PO localization files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/txt2po.html for examples and usage instructions.

```
translate.convert.txt2po.run_converter(input_file, output_file, template_file=None, duplicat-
                                                  estyle='msgctxt', encoding='utf-8', flavour=None,
                                                  no segmentation=False)
     Wrapper around converter.
class translate.convert.txt2po.txt2po(input_file,
                                                               output_file,
                                                                              template_file=None,
                                                 duplicate style='msgctxt',
                                                                                 encoding='utf-8',
                                                 flavour=None, no_segmentation=False)
     Convert one plain text (.txt) file to a single PO file.
     SourceStoreClass
          alias of translate.storage.txt.TxtFile
     TargetStoreClass
          alias of translate.storage.pypo.pofile
     TargetUnitClass
          alias of translate.storage.pypo.pounit
     convert store()
          Convert a single source format file to a target format file.
     merge_stores()
          Convert two source format files to a target format file.
     run()
          Run the converter.
```

# web2py2po

Convert web2py translation dictionaries (.py) to GNU/gettext PO files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/web2py2po.html for examples and usage instructions.

#### xliff2odf

Convert XLIFF translation files to OpenDocument (ODF) files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/odf2xliff.html for examples and usage instructions.

```
translate.convert.xliff2odf.convertxliff(input_file, output_file, template)
    Create a translated ODF using an ODF template and a XLIFF file.
translate.convert.xliff2odf.write_odf(template, output_file, dom_trees)
    Write the translated ODF package.
```

The resulting ODF package is a copy of the template ODF package, with the translatable files replaced by their translated versions.

#### xliff2oo

Convert XLIFF localization files to an OpenOffice.org (SDF) localization file.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/oo2po.html for examples and usage instructions.

# xliff2po

Convert XLIFF localization files to Gettext PO localization files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/xliff2po.html for examples and usage instructions.

```
translate.convert.xliff2po.convertxliff(inputfile, outputfile, templates, duplicat-
estyle='msgctxt')
reads in stdin using fromfileclass, converts using convertorclass, writes to stdout
```

# yaml2po

Convert YAML files to Gettext PO localization files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/yaml2po.html for examples and usage instructions.

```
translate.convert.yaml2po.run_converter(input_file,
                                                                output_file,
                                                                              template_file=None,
                                                   pot=False, duplicatestyle='msgctxt')
     Wrapper around converter.
class translate.convert.yaml2po.yaml2po(input_file,
                                                                              template_file=None,
                                                                output_file,
                                                   blank_msgstr=False, duplicate_style='msgctxt')
     Convert one or two YAML files to a single PO file.
     SourceStoreClass
          alias of translate.storage.yaml.YAMLFile
     TargetStoreClass
          alias of translate.storage.pypo.pofile
     TargetUnitClass
          alias of translate.storage.pypo.pounit
     convert store()
          Convert a single source format file to a target format file.
     convert unit(unit)
          Convert a source format unit to a target format unit.
     merge_stores()
          Convert two source format files to a target format file.
     run()
          Run the converter.
```

#### filters

Filters that can be used on translations...

### autocorrect

A set of autocorrect functions that fix common punctuation and space problems automatically

```
translate.filters.autocorrect.correct(source, target)
Runs a set of easy and automatic corrections
```

# **Current corrections include:**

- Ellipses align target to use source form of ellipses (either three dots or the Unicode ellipses characters)
- Missing whitespace and start or end of the target
- Missing punction (.:?) at the end of the target

#### checks

This is a set of validation checks that can be performed on translation units.

Derivatives of UnitChecker (like StandardUnitChecker) check translation units, and derivatives of TranslationChecker (like StandardChecker) check (source, target) translation pairs.

When adding a new test here, please document and explain their behaviour on the *pofilter tests* page.

class translate.filters.checks.CCLicenseChecker(\*\*kwargs)

```
accelerators (*args, **kwargs)
```

Checks whether accelerators are consistent between the two strings.

This test is capable of checking the different type of accelerators that are used in different projects, like Mozilla or KDE. The test will pick up accelerators that are missing and ones that shouldn't be there.

See accelerators on the localization guide for a full description on accelerators.

```
acronyms (*args, **kwargs)
```

Checks that acronyms that appear are unchanged.

If an acronym appears in the original this test will check that it appears in the translation. Translating acronyms is a language decision but many languages leave them unchanged. In that case this test is useful for tracking down translations of the acronym and correcting them.

```
blank (*args, **kwargs)
```

Checks whether a translation is totally blank.

This will check to see if a translation has inadvertently been translated as blank i.e. as spaces. This is different from untranslated which is completely empty. This test is useful in that if something is translated as "" it will appear to most tools as if it is translated.

```
brackets (*args, **kwargs)
```

Checks that the number of brackets in both strings match.

If ([{ or }]) appear in the original this will check that the same number appear in the translation.

#### checker\_name

Extract checker name, for example 'mozilla' from MozillaChecker.

```
compendiumconflicts(*args, **kwargs)
```

Checks for Gettext compendium conflicts (#-#-#-#).

When you use msgcat to create a PO compendium it will insert #-#-#-# into entries that are not consistent. If the compendium is used later in a message merge then these conflicts will appear in your translations. This test quickly extracts those for correction.

```
credits(*args, **kwargs)
```

Checks for messages containing translation credits instead of normal translations.

Some projects have consistent ways of giving credit to translators by having a unit or two where translators can fill in their name and possibly their contact details. This test allows you to find these units easily to

check that they are completed correctly and also disables other tests that might incorrectly get triggered for these units (such as urls, emails, etc.)

## doublequoting(\*args, \*\*kwargs)

Checks whether doublequoting is consistent between the two strings.

Checks on double quotes " to ensure that you have the same number in both the original and the translated string. This tests takes into account that several languages use different quoting characters, and will test for them instead.

# doublespacing(\*args, \*\*kwargs)

Checks for bad double-spaces by comparing to original.

This will identify if you have [space] [space] in when you don't have it in the original or it appears in the original but not in your translation. Some of these are spurious and how you correct them depends on the conventions of your language.

# doublewords (\*args, \*\*kwargs)

Checks for repeated words in the translation.

Words that have been repeated in a translation will be highlighted with this test e.g. "the the", "a a". These are generally typos that need correcting. Some languages may have valid repeated words in their structure, in that case either ignore those instances or switch this test off.

# emails (\*args, \*\*kwargs)

Checks that emails are not translated.

Generally you should not be translating email addresses. This check will look to see that email addresses e.g. info@example.com are not translated. In some cases of course you should translate the address but generally you shouldn't.

# endpunc (\*args, \*\*kwargs)

Checks whether punctuation at the end of the strings match.

This will ensure that the ending of your translation has the same punctuation as the original. E.g. if it ends in :[space] then so should yours. It is useful for ensuring that you have ellipses [...] in all your translations, not simply three separate full-stops. You may pick up some errors in the original: feel free to keep your translation and notify the programmers. In some languages, characters such as ? or ! are always preceded by a space e.g. [space]? — do what your language customs dictate. Other false positives you will notice are, for example, if through changes in word-order you add "), etc. at the end of the sentence. Do not change these: your language word-order takes precedence.

It must be noted that if you are tempted to leave out [full-stop] or [colon] or add [full-stop] to a sentence, that often these have been done for a reason, e.g. a list where fullstops make it look cluttered. So, initially match them with the English, and make changes once the program is being used.

This check is aware of several language conventions for punctuation characters, such as the custom question marks for Greek and Arabic, Devanagari Danda, full-width punctuation for CJK languages, etc. Support for your language can be added easily if it is not there yet.

# endwhitespace(\*args, \*\*kwargs)

Checks whether whitespace at the end of the strings matches.

Operates the same as endpunc but is only concerned with whitespace. This filter is particularly useful for those strings which will evidently be followed by another string in the program, e.g. [Password: ] or [Enter your username: ]. The whitespace is an inherent part of the string. This filter makes sure you don't miss those important but otherwise invisible spaces!

If your language uses full-width punctuation (like Chinese), the visual spacing in the character might be enough without an added extra space.

### escapes (\*args, \*\*kwargs)

Checks whether escaping is consistent between the two strings.

Checks escapes such as  $\n \u0000$  to ensure that if they exist in the original string you also have them in the translation.

# filepaths (\*args, \*\*kwargs)

Checks that file paths have not been translated.

Checks that paths such as /home/user1 have not been translated. Generally you do not translate a file path, unless it is being used as an example, e.g. your\_user\_name/path/to/filename.conf.

# filteraccelerators\_by\_list(str1, acceptlist=None)

Filter out accelerators from str1.

# functions (\*args, \*\*kwargs)

Checks that function names are not translated.

Checks that function names e.g. rgb() or getEntity.Name() are not translated.

#### get\_ignored\_filters()

Return checker's additional filters for current language.

#### getfilters (excludefilters=None, limitfilters=None)

Returns dictionary of available filters, including/excluding those in the given lists.

## kdecomments (\*args, \*\*kwargs)

Checks to ensure that no KDE style comments appear in the translation.

KDE style translator comments appear in PO files as "\_: comment\n". New translators often translate the comment. This test tries to identify instances where the comment has been translated.

### long (\*args, \*\*kwargs)

Checks whether a translation is much longer than the original string.

This is most useful in the special case where the translation is multiple characters long while the source text is only 1 character long. Otherwise, we use a general ratio that will catch very big differences but is set conservatively to limit the number of false positives.

## musttranslatewords (\*args, \*\*kwargs)

Checks that words configured as definitely translatable don't appear in the translation.

If for instance in your language you decide that you must translate 'OK' then this test will flag any occurrences of 'OK' in the translation if it appeared in the source string. You must specify a file containing all of the *must translate* words using --musttranslatefile.

## newlines (\*args, \*\*kwargs)

Checks whether newlines are consistent between the two strings.

Counts the number of  $\n$  newlines (and variants such as  $\r$ ) and reports and error if they differ.

# notranslatewords (\*args, \*\*kwargs)

Checks that words configured as untranslatable appear in the translation too.

Many brand names should not be translated, this test allows you to easily make sure that words like: Word, Excel, Impress, Calc, etc. are not translated. You must specify a file containing all of the *no translate* words using --notranslatefile.

# numbers (\*args, \*\*kwargs)

Checks whether numbers of various forms are consistent between the two strings.

You will see some errors where you have either written the number in full or converted it to the digit in your translation. Also changes in order will trigger this error.

#### options (\*args, \*\*kwargs)

Checks that command line options are not translated.

In messages that contain command line options, such as <code>--help</code>, this test will check that these remain untranslated. These could be translated in the future if programs can create a mechanism to allow this, but currently they are not translated. If the options has a parameter, e.g. <code>--file=FILE</code>, then the test will check that the parameter has been translated.

# printf(\*args, \*\*kwargs)

Checks whether printf format strings match.

If the printf formatting variables are not identical, then this will indicate an error. Printf statements are used by programs to format output in a human readable form (they are placeholders for variable data). They allow you to specify lengths of string variables, string padding, number padding, precision, etc. Generally they will look like this: %d, %5.2f, %100s, etc. The test can also manage variables-reordering using the %1\$s syntax. The variables' type and details following data are tested to ensure that they are strictly identical, but they may be reordered.

See also printf Format String.

# puncspacing(\*args, \*\*kwargs)

Checks for bad spacing after punctuation.

In the case of [full-stop][space] in the original, this test checks that your translation does not remove the space. It checks also for [comma], [colon], etc.

Some languages don't use spaces after common punctuation marks, especially where full-width punctuation marks are used. This check will take that into account.

### purepunc (\*args, \*\*kwargs)

Checks that strings that are purely punctuation are not changed.

This extracts strings like + or - as these usually should not be changed.

# pythonbraceformat (\*args, \*\*kwargs)

Checks whether python brace format strings match.

#### run\_filters (unit, categorised=False)

Do some optimisation by caching some data of the unit for the benefit of run\_test().

#### run\_test (test, unit)

Runs the given test on the given unit.

Note that this can raise a *FilterFailure* as part of normal operation.

## sentencecount (\*args, \*\*kwargs)

Checks that the number of sentences in both strings match.

Adds the number of sentences to see that the sentence count is the same between the original and translated string. You may not always want to use this test, if you find you often need to reformat your translation, because the original is badly-expressed, or because the structure of your language works better that way. Do what works best for your language: it's the meaning of the original you want to convey, not the exact way it was written in the English.

#### setconfig(config)

Sets the accelerator list.

#### setsuggestionstore(store)

Sets the filename that a checker should use for evaluating suggestions.

```
short (*args, **kwargs)
```

Checks whether a translation is much shorter than the original string.

This is most useful in the special case where the translation is 1 characters long while the source text is multiple characters long. Otherwise, we use a general ratio that will catch very big differences but is set conservatively to limit the number of false positives.

## simplecaps (\*args, \*\*kwargs)

Checks the capitalisation of two strings isn't wildly different.

This will pick up many false positives, so don't be a slave to it. It is useful for identifying translations that don't start with a capital letter (upper-case letter) when they should, or those that do when they shouldn't. It will also highlight sentences that have extra capitals; depending on the capitalisation convention of your language, you might want to change these to Title Case, or change them all to normal sentence case.

# simpleplurals(\*args, \*\*kwargs)

Checks for English style plural(s) for you to review.

This test will extract any message that contains words with a final "(s)" in the source text. You can then inspect the message, to check that the correct plural form has been used for your language. In some languages, plurals are made by adding text at the beginning of words, making the English style messy. In this case, they often revert to the plural form. This test allows an editor to check that the plurals used are correct. Be aware that this test may create a number of false positives.

For languages with no plural forms (only one noun form) this test will simply test that nothing like "(s)" was used in the translation.

### singlequoting(\*args, \*\*kwargs)

Checks whether singlequoting is consistent between the two strings.

The same as doublequoting but checks for the 'character. Because this is used in contractions like it's and in possessive forms like user's, this test can output spurious errors if your language doesn't use such forms. If a quote appears at the end of a sentence in the translation, i.e. '., this might not be detected properly by the check.

# spellcheck (\*args, \*\*kwargs)

Checks words that don't pass a spell check.

This test will check for misspelled words in your translation. The test first checks for misspelled words in the original (usually English) text, and adds those to an exclusion list. The advantage of this exclusion is that many words that are specific to the application will not raise errors e.g. program names, brand names, function names.

The checker works with PyEnchant. You need to have PyEnchant installed as well as a dictionary for your language (for example, one of the Hunspell or aspell dictionaries). This test will only work if you have specified the --language option.

The pofilter error that is created, lists the misspelled word, plus suggestions returned from the spell checker. That makes it easy for you to identify the word and select a replacement.

### startcaps (\*args, \*\*kwargs)

Checks that the message starts with the correct capitalisation.

After stripping whitespace and common punctuation characters, it then checks to see that the first remaining character is correctly capitalised. So, if the sentence starts with an upper-case letter, and the translation does not, an error is produced.

This check is entirely disabled for many languages that don't make a distinction between upper and lower case. Contact us if this is not yet disabled for your language.

# startpunc(\*args, \*\*kwargs)

Checks whether punctuation at the beginning of the strings match.

Operates as endpunc but you will probably see fewer errors.

#### startwhitespace (\*args, \*\*kwargs)

Checks whether whitespace at the beginning of the strings matches.

As in endwhitespace but you will see fewer errors.

#### tabs (\*args, \*\*kwargs)

Checks whether tabs are consistent between the two strings.

Counts the number of \t tab markers and reports an error if they differ.

## unchanged(\*args, \*\*kwargs)

Checks whether a translation is basically identical to the original string.

This checks to see if the translation isn't just a copy of the English original. Sometimes, this is what you want, but other times you will detect words that should have been translated.

#### untranslated(\*args, \*\*kwargs)

Checks whether a string has been translated at all.

This check is really only useful if you want to extract untranslated strings so that they can be translated independently of the main work.

### urls (\*args, \*\*kwargs)

Checks that URLs are not translated.

This checks only basic URLs (http, ftp, mailto etc.) not all URIs (e.g. afp, smb, file). Generally, you don't want to translate URLs, unless they are example URLs (http://your\_server.com/filename.html). If the URL is for configuration information, then you need to query the developers about placing configuration information in PO files. It shouldn't really be there, unless it is very clearly marked: such information should go into a configuration file.

## validchars (\*args, \*\*kwargs)

Checks that only characters specified as valid appear in the translation.

Often during character conversion to and from UTF-8 you get some strange characters appearing in your translation. This test presents a simple way to try and identify such errors.

This test will only run of you specify the --validcharsfile command line option. This file contains all the characters that are valid in your language. You must use UTF-8 encoding for the characters in the file.

If the test finds any characters not in your valid characters file then the test will print the character together with its Unicode value (e.g. 002B).

# variables (\*args, \*\*kwargs)

Checks whether variables of various forms are consistent between the two strings.

This checks to make sure that variables that appear in the original also appear in the translation. It can handle variables from projects like KDE or OpenOffice. It does not at the moment cope with variables that use the reordering syntax of Gettext PO files.

# xmltags (\*args, \*\*kwargs)

Checks that XML/HTML tags have not been translated.

This check finds the number of tags in the source string and checks that the same number are in the translation. If the counts don't match then either the tag is missing or it was mistakenly translated by the translator, both of which are errors.

The check ignores tags or things that look like tags that cover the whole string e.g. <Error> but will produce false positives for things like An <Error> occurred as here Error should be translated. It also will allow translation of the *alt* attribute in e.g. <img src="bob.png" alt="Image description"> or similar translatable attributes in OpenOffice.org help files.

class translate.filters.checks.CheckerConfig(targetlanguage=None, accelmark-

ers=None, varmatches=None, notranslatewords=None, musttranslatewords=None, validchars=None, punctuation=None, endpunctuation=None, ignoretags=None, canchangetags=None, criticaltests=None, credit\_sources=None)

Object representing the configuration of a checker.

## update (otherconfig)

Combines the info in otherconfig into this config object.

# ${\tt updatetargetlanguage}\ (langcode)$

Updates the target language in the config to the given target language and sets its script.

# updatevalidchars (validchars)

Updates the map that eliminates valid characters.

class translate.filters.checks.DrupalChecker(\*\*kwargs)

```
accelerators (*args, **kwargs)
```

Checks whether accelerators are consistent between the two strings.

This test is capable of checking the different type of accelerators that are used in different projects, like Mozilla or KDE. The test will pick up accelerators that are missing and ones that shouldn't be there.

See accelerators on the localization guide for a full description on accelerators.

```
acronyms (*args, **kwargs)
```

Checks that acronyms that appear are unchanged.

If an acronym appears in the original this test will check that it appears in the translation. Translating acronyms is a language decision but many languages leave them unchanged. In that case this test is useful for tracking down translations of the acronym and correcting them.

```
blank (*args, **kwargs)
```

Checks whether a translation is totally blank.

This will check to see if a translation has inadvertently been translated as blank i.e. as spaces. This is different from untranslated which is completely empty. This test is useful in that if something is translated as "" it will appear to most tools as if it is translated.

```
brackets (*args, **kwargs)
```

Checks that the number of brackets in both strings match.

If ([{ or }]) appear in the original this will check that the same number appear in the translation.

#### checker\_name

Extract checker name, for example 'mozilla' from MozillaChecker.

### compendiumconflicts(\*args, \*\*kwargs)

Checks for Gettext compendium conflicts (#-#-#-#).

When you use msgcat to create a PO compendium it will insert #-#-#-# into entries that are not consistent. If the compendium is used later in a message merge then these conflicts will appear in your translations. This test quickly extracts those for correction.

```
credits (*args, **kwargs)
```

Checks for messages containing translation credits instead of normal translations.

Some projects have consistent ways of giving credit to translators by having a unit or two where translators can fill in their name and possibly their contact details. This test allows you to find these units easily to

check that they are completed correctly and also disables other tests that might incorrectly get triggered for these units (such as urls, emails, etc.)

## doublequoting(\*args, \*\*kwargs)

Checks whether doublequoting is consistent between the two strings.

Checks on double quotes " to ensure that you have the same number in both the original and the translated string. This tests takes into account that several languages use different quoting characters, and will test for them instead.

# doublespacing(\*args, \*\*kwargs)

Checks for bad double-spaces by comparing to original.

This will identify if you have [space] [space] in when you don't have it in the original or it appears in the original but not in your translation. Some of these are spurious and how you correct them depends on the conventions of your language.

# doublewords (\*args, \*\*kwargs)

Checks for repeated words in the translation.

Words that have been repeated in a translation will be highlighted with this test e.g. "the the", "a a". These are generally typos that need correcting. Some languages may have valid repeated words in their structure, in that case either ignore those instances or switch this test off.

# emails (\*args, \*\*kwargs)

Checks that emails are not translated.

Generally you should not be translating email addresses. This check will look to see that email addresses e.g. info@example.com are not translated. In some cases of course you should translate the address but generally you shouldn't.

# endpunc (\*args, \*\*kwargs)

Checks whether punctuation at the end of the strings match.

This will ensure that the ending of your translation has the same punctuation as the original. E.g. if it ends in :[space] then so should yours. It is useful for ensuring that you have ellipses [...] in all your translations, not simply three separate full-stops. You may pick up some errors in the original: feel free to keep your translation and notify the programmers. In some languages, characters such as ? or ! are always preceded by a space e.g. [space]? — do what your language customs dictate. Other false positives you will notice are, for example, if through changes in word-order you add "), etc. at the end of the sentence. Do not change these: your language word-order takes precedence.

It must be noted that if you are tempted to leave out [full-stop] or [colon] or add [full-stop] to a sentence, that often these have been done for a reason, e.g. a list where fullstops make it look cluttered. So, initially match them with the English, and make changes once the program is being used.

This check is aware of several language conventions for punctuation characters, such as the custom question marks for Greek and Arabic, Devanagari Danda, full-width punctuation for CJK languages, etc. Support for your language can be added easily if it is not there yet.

# endwhitespace(\*args, \*\*kwargs)

Checks whether whitespace at the end of the strings matches.

Operates the same as endpunc but is only concerned with whitespace. This filter is particularly useful for those strings which will evidently be followed by another string in the program, e.g. [Password: ] or [Enter your username: ]. The whitespace is an inherent part of the string. This filter makes sure you don't miss those important but otherwise invisible spaces!

If your language uses full-width punctuation (like Chinese), the visual spacing in the character might be enough without an added extra space.

### escapes (\*args, \*\*kwargs)

Checks whether escaping is consistent between the two strings.

Checks escapes such as  $\n \u0000$  to ensure that if they exist in the original string you also have them in the translation.

# filepaths (\*args, \*\*kwargs)

Checks that file paths have not been translated.

Checks that paths such as /home/user1 have not been translated. Generally you do not translate a file path, unless it is being used as an example, e.g. your\_user\_name/path/to/filename.conf.

# filteraccelerators\_by\_list(str1, acceptlist=None)

Filter out accelerators from str1.

# functions (\*args, \*\*kwargs)

Checks that function names are not translated.

Checks that function names e.g. rgb() or getEntity.Name() are not translated.

#### get\_ignored\_filters()

Return checker's additional filters for current language.

# getfilters (excludefilters=None, limitfilters=None)

Returns dictionary of available filters, including/excluding those in the given lists.

## kdecomments (\*args, \*\*kwargs)

Checks to ensure that no KDE style comments appear in the translation.

KDE style translator comments appear in PO files as "\_: comment\n". New translators often translate the comment. This test tries to identify instances where the comment has been translated.

### long (\*args, \*\*kwargs)

Checks whether a translation is much longer than the original string.

This is most useful in the special case where the translation is multiple characters long while the source text is only 1 character long. Otherwise, we use a general ratio that will catch very big differences but is set conservatively to limit the number of false positives.

## musttranslatewords (\*args, \*\*kwargs)

Checks that words configured as definitely translatable don't appear in the translation.

If for instance in your language you decide that you must translate 'OK' then this test will flag any occurrences of 'OK' in the translation if it appeared in the source string. You must specify a file containing all of the *must translate* words using --musttranslatefile.

## newlines (\*args, \*\*kwargs)

Checks whether newlines are consistent between the two strings.

Counts the number of  $\n$  newlines (and variants such as  $\n$ ) and reports and error if they differ.

# notranslatewords (\*args, \*\*kwargs)

Checks that words configured as untranslatable appear in the translation too.

Many brand names should not be translated, this test allows you to easily make sure that words like: Word, Excel, Impress, Calc, etc. are not translated. You must specify a file containing all of the *no translate* words using --notranslatefile.

# numbers (\*args, \*\*kwargs)

Checks whether numbers of various forms are consistent between the two strings.

You will see some errors where you have either written the number in full or converted it to the digit in your translation. Also changes in order will trigger this error.

```
options (*args, **kwargs)
```

Checks that command line options are not translated.

In messages that contain command line options, such as <code>--help</code>, this test will check that these remain untranslated. These could be translated in the future if programs can create a mechanism to allow this, but currently they are not translated. If the options has a parameter, e.g. <code>--file=FILE</code>, then the test will check that the parameter has been translated.

```
printf(*args, **kwargs)
```

Checks whether printf format strings match.

If the printf formatting variables are not identical, then this will indicate an error. Printf statements are used by programs to format output in a human readable form (they are placeholders for variable data). They allow you to specify lengths of string variables, string padding, number padding, precision, etc. Generally they will look like this: %d, %5.2f, %100s, etc. The test can also manage variables-reordering using the %1\$s syntax. The variables' type and details following data are tested to ensure that they are strictly identical, but they may be reordered.

See also printf Format String.

# puncspacing(\*args, \*\*kwargs)

Checks for bad spacing after punctuation.

In the case of [full-stop][space] in the original, this test checks that your translation does not remove the space. It checks also for [comma], [colon], etc.

Some languages don't use spaces after common punctuation marks, especially where full-width punctuation marks are used. This check will take that into account.

### purepunc (\*args, \*\*kwargs)

Checks that strings that are purely punctuation are not changed.

This extracts strings like + or - as these usually should not be changed.

# pythonbraceformat (\*args, \*\*kwargs)

Checks whether python brace format strings match.

```
run_filters (unit, categorised=False)
```

Do some optimisation by caching some data of the unit for the benefit of run\_test().

```
run_test (test, unit)
```

Runs the given test on the given unit.

Note that this can raise a *FilterFailure* as part of normal operation.

```
sentencecount (*args, **kwargs)
```

Checks that the number of sentences in both strings match.

Adds the number of sentences to see that the sentence count is the same between the original and translated string. You may not always want to use this test, if you find you often need to reformat your translation, because the original is badly-expressed, or because the structure of your language works better that way. Do what works best for your language: it's the meaning of the original you want to convey, not the exact way it was written in the English.

#### setconfig(config)

Sets the accelerator list.

#### setsuggestionstore(store)

Sets the filename that a checker should use for evaluating suggestions.

```
short (*args, **kwargs)
```

Checks whether a translation is much shorter than the original string.

This is most useful in the special case where the translation is 1 characters long while the source text is multiple characters long. Otherwise, we use a general ratio that will catch very big differences but is set conservatively to limit the number of false positives.

## simplecaps (\*args, \*\*kwargs)

Checks the capitalisation of two strings isn't wildly different.

This will pick up many false positives, so don't be a slave to it. It is useful for identifying translations that don't start with a capital letter (upper-case letter) when they should, or those that do when they shouldn't. It will also highlight sentences that have extra capitals; depending on the capitalisation convention of your language, you might want to change these to Title Case, or change them all to normal sentence case.

# simpleplurals(\*args, \*\*kwargs)

Checks for English style plural(s) for you to review.

This test will extract any message that contains words with a final "(s)" in the source text. You can then inspect the message, to check that the correct plural form has been used for your language. In some languages, plurals are made by adding text at the beginning of words, making the English style messy. In this case, they often revert to the plural form. This test allows an editor to check that the plurals used are correct. Be aware that this test may create a number of false positives.

For languages with no plural forms (only one noun form) this test will simply test that nothing like "(s)" was used in the translation.

### singlequoting(\*args, \*\*kwargs)

Checks whether singlequoting is consistent between the two strings.

The same as doublequoting but checks for the 'character. Because this is used in contractions like it's and in possessive forms like user's, this test can output spurious errors if your language doesn't use such forms. If a quote appears at the end of a sentence in the translation, i.e. '., this might not be detected properly by the check.

# spellcheck (\*args, \*\*kwargs)

Checks words that don't pass a spell check.

This test will check for misspelled words in your translation. The test first checks for misspelled words in the original (usually English) text, and adds those to an exclusion list. The advantage of this exclusion is that many words that are specific to the application will not raise errors e.g. program names, brand names, function names.

The checker works with PyEnchant. You need to have PyEnchant installed as well as a dictionary for your language (for example, one of the Hunspell or aspell dictionaries). This test will only work if you have specified the --language option.

The pofilter error that is created, lists the misspelled word, plus suggestions returned from the spell checker. That makes it easy for you to identify the word and select a replacement.

### startcaps (\*args, \*\*kwargs)

Checks that the message starts with the correct capitalisation.

After stripping whitespace and common punctuation characters, it then checks to see that the first remaining character is correctly capitalised. So, if the sentence starts with an upper-case letter, and the translation does not, an error is produced.

This check is entirely disabled for many languages that don't make a distinction between upper and lower case. Contact us if this is not yet disabled for your language.

# startpunc(\*args, \*\*kwargs)

Checks whether punctuation at the beginning of the strings match.

Operates as endpunc but you will probably see fewer errors.

#### startwhitespace (\*args, \*\*kwargs)

Checks whether whitespace at the beginning of the strings matches.

As in endwhitespace but you will see fewer errors.

#### tabs (\*args, \*\*kwargs)

Checks whether tabs are consistent between the two strings.

Counts the number of \t tab markers and reports an error if they differ.

## unchanged(\*args, \*\*kwargs)

Checks whether a translation is basically identical to the original string.

This checks to see if the translation isn't just a copy of the English original. Sometimes, this is what you want, but other times you will detect words that should have been translated.

### untranslated(\*args, \*\*kwargs)

Checks whether a string has been translated at all.

This check is really only useful if you want to extract untranslated strings so that they can be translated independently of the main work.

### urls (\*args, \*\*kwargs)

Checks that URLs are not translated.

This checks only basic URLs (http, ftp, mailto etc.) not all URIs (e.g. afp, smb, file). Generally, you don't want to translate URLs, unless they are example URLs (http://your\_server.com/filename.html). If the URL is for configuration information, then you need to query the developers about placing configuration information in PO files. It shouldn't really be there, unless it is very clearly marked: such information should go into a configuration file.

## validchars (\*args, \*\*kwargs)

Checks that only characters specified as valid appear in the translation.

Often during character conversion to and from UTF-8 you get some strange characters appearing in your translation. This test presents a simple way to try and identify such errors.

This test will only run of you specify the --validcharsfile command line option. This file contains all the characters that are valid in your language. You must use UTF-8 encoding for the characters in the file.

If the test finds any characters not in your valid characters file then the test will print the character together with its Unicode value (e.g. 002B).

# variables (\*args, \*\*kwargs)

Checks whether variables of various forms are consistent between the two strings.

This checks to make sure that variables that appear in the original also appear in the translation. It can handle variables from projects like KDE or OpenOffice. It does not at the moment cope with variables that use the reordering syntax of Gettext PO files.

# xmltags (\*args, \*\*kwargs)

Checks that XML/HTML tags have not been translated.

This check finds the number of tags in the source string and checks that the same number are in the translation. If the counts don't match then either the tag is missing or it was mistakenly translated by the translator, both of which are errors.

The check ignores tags or things that look like tags that cover the whole string e.g. <Error> but will produce false positives for things like An <Error> occurred as here Error should be translated. It also will allow translation of the *alt* attribute in e.g. <img src="bob.png" alt="Image description"> or similar translatable attributes in OpenOffice.org help files.

### exception translate.filters.checks.FilterFailure(messages)

This exception signals that a Filter didn't pass, and gives an explanation or a comment.

class translate.filters.checks.GnomeChecker(\*\*kwargs)

#### accelerators (\*args, \*\*kwargs)

Checks whether accelerators are consistent between the two strings.

This test is capable of checking the different type of accelerators that are used in different projects, like Mozilla or KDE. The test will pick up accelerators that are missing and ones that shouldn't be there.

See accelerators on the localization guide for a full description on accelerators.

```
acronyms (*args, **kwargs)
```

Checks that acronyms that appear are unchanged.

If an acronym appears in the original this test will check that it appears in the translation. Translating acronyms is a language decision but many languages leave them unchanged. In that case this test is useful for tracking down translations of the acronym and correcting them.

#### blank (\*args, \*\*kwargs)

Checks whether a translation is totally blank.

This will check to see if a translation has inadvertently been translated as blank i.e. as spaces. This is different from untranslated which is completely empty. This test is useful in that if something is translated as "" it will appear to most tools as if it is translated.

## brackets (\*args, \*\*kwargs)

Checks that the number of brackets in both strings match.

If ([{ or }]) appear in the original this will check that the same number appear in the translation.

# checker\_name

Extract checker name, for example 'mozilla' from MozillaChecker.

# compendiumconflicts(\*args, \*\*kwargs)

Checks for Gettext compendium conflicts (#-#-#-#).

When you use msgcat to create a PO compendium it will insert #-#-#-# into entries that are not consistent. If the compendium is used later in a message merge then these conflicts will appear in your translations. This test quickly extracts those for correction.

```
credits(*args, **kwargs)
```

Checks for messages containing translation credits instead of normal translations.

Some projects have consistent ways of giving credit to translators by having a unit or two where translators can fill in their name and possibly their contact details. This test allows you to find these units easily to check that they are completed correctly and also disables other tests that might incorrectly get triggered for these units (such as urls, emails, etc.)

# doublequoting(\*args, \*\*kwargs)

Checks whether doublequoting is consistent between the two strings.

Checks on double quotes " to ensure that you have the same number in both the original and the translated string. This tests takes into account that several languages use different quoting characters, and will test for them instead.

### doublespacing(\*args, \*\*kwargs)

Checks for bad double-spaces by comparing to original.

This will identify if you have [space][space] in when you don't have it in the original or it appears in the original but not in your translation. Some of these are spurious and how you correct them depends on the conventions of your language.

## doublewords (\*args, \*\*kwargs)

Checks for repeated words in the translation.

Words that have been repeated in a translation will be highlighted with this test e.g. "the the", "a a". These are generally typos that need correcting. Some languages may have valid repeated words in their structure, in that case either ignore those instances or switch this test off.

## emails (\*args, \*\*kwargs)

Checks that emails are not translated.

Generally you should not be translating email addresses. This check will look to see that email addresses e.g. info@example.com are not translated. In some cases of course you should translate the address but generally you shouldn't.

# endpunc (\*args, \*\*kwargs)

Checks whether punctuation at the end of the strings match.

This will ensure that the ending of your translation has the same punctuation as the original. E.g. if it ends in :[space] then so should yours. It is useful for ensuring that you have ellipses [...] in all your translations, not simply three separate full-stops. You may pick up some errors in the original: feel free to keep your translation and notify the programmers. In some languages, characters such as ? or ! are always preceded by a space e.g. [space]? — do what your language customs dictate. Other false positives you will notice are, for example, if through changes in word-order you add "), etc. at the end of the sentence. Do not change these: your language word-order takes precedence.

It must be noted that if you are tempted to leave out [full-stop] or [colon] or add [full-stop] to a sentence, that often these have been done for a reason, e.g. a list where fullstops make it look cluttered. So, initially match them with the English, and make changes once the program is being used.

This check is aware of several language conventions for punctuation characters, such as the custom question marks for Greek and Arabic, Devanagari Danda, full-width punctuation for CJK languages, etc. Support for your language can be added easily if it is not there yet.

## endwhitespace(\*args, \*\*kwargs)

Checks whether whitespace at the end of the strings matches.

Operates the same as endpunc but is only concerned with whitespace. This filter is particularly useful for those strings which will evidently be followed by another string in the program, e.g. [Password: ] or [Enter your username: ]. The whitespace is an inherent part of the string. This filter makes sure you don't miss those important but otherwise invisible spaces!

If your language uses full-width punctuation (like Chinese), the visual spacing in the character might be enough without an added extra space.

## escapes (\*args, \*\*kwargs)

Checks whether escaping is consistent between the two strings.

Checks escapes such as  $\n \u0000$  to ensure that if they exist in the original string you also have them in the translation.

#### filepaths (\*args, \*\*kwargs)

Checks that file paths have not been translated.

Checks that paths such as /home/user1 have not been translated. Generally you do not translate a file path, unless it is being used as an example, e.g. your\_user\_name/path/to/filename.conf.

# filteraccelerators\_by\_list(str1, acceptlist=None)

Filter out accelerators from str1.

### functions (\*args, \*\*kwargs)

Checks that function names are not translated.

Checks that function names e.g. rgb() or getEntity.Name() are not translated.

## gconf (\*args, \*\*kwargs)

Checks if we have any gconf config settings translated.

Gconf settings should not be translated so this check checks that gconf settings such as "name" or "modification\_date" are not translated in the translation. It allows you to change the surrounding quotes but will ensure that the setting values remain untranslated.

## get\_ignored\_filters()

Return checker's additional filters for current language.

# getfilters (excludefilters=None, limitfilters=None)

Returns dictionary of available filters, including/excluding those in the given lists.

### kdecomments (\*args, \*\*kwargs)

Checks to ensure that no KDE style comments appear in the translation.

KDE style translator comments appear in PO files as "\_: comment\n". New translators often translate the comment. This test tries to identify instances where the comment has been translated.

# long(\*args, \*\*kwargs)

Checks whether a translation is much longer than the original string.

This is most useful in the special case where the translation is multiple characters long while the source text is only 1 character long. Otherwise, we use a general ratio that will catch very big differences but is set conservatively to limit the number of false positives.

## musttranslatewords (\*args, \*\*kwargs)

Checks that words configured as definitely translatable don't appear in the translation.

If for instance in your language you decide that you must translate 'OK' then this test will flag any occurrences of 'OK' in the translation if it appeared in the source string. You must specify a file containing all of the *must translate* words using --musttranslatefile.

```
newlines (*args, **kwargs)
```

Checks whether newlines are consistent between the two strings.

Counts the number of \n newlines (and variants such as \r\n) and reports and error if they differ.

## notranslatewords (\*args, \*\*kwargs)

Checks that words configured as untranslatable appear in the translation too.

Many brand names should not be translated, this test allows you to easily make sure that words like: Word, Excel, Impress, Calc, etc. are not translated. You must specify a file containing all of the *no translate* words using --notranslatefile.

```
numbers (*args, **kwargs)
```

Checks whether numbers of various forms are consistent between the two strings.

You will see some errors where you have either written the number in full or converted it to the digit in your translation. Also changes in order will trigger this error.

#### options (\*args, \*\*kwargs)

Checks that command line options are not translated.

In messages that contain command line options, such as <code>--help</code>, this test will check that these remain untranslated. These could be translated in the future if programs can create a mechanism to allow this, but currently they are not translated. If the options has a parameter, e.g. <code>--file=FILE</code>, then the test will check that the parameter has been translated.

#### printf(\*args, \*\*kwargs)

Checks whether printf format strings match.

If the printf formatting variables are not identical, then this will indicate an error. Printf statements are used by programs to format output in a human readable form (they are placeholders for variable data). They allow you to specify lengths of string variables, string padding, number padding, precision, etc. Generally they will look like this: %d, %5.2f, %100s, etc. The test can also manage variables-reordering using the %1\$s syntax. The variables' type and details following data are tested to ensure that they are strictly identical, but they may be reordered.

See also printf Format String.

# puncspacing(\*args, \*\*kwargs)

Checks for bad spacing after punctuation.

In the case of [full-stop][space] in the original, this test checks that your translation does not remove the space. It checks also for [comma], [colon], etc.

Some languages don't use spaces after common punctuation marks, especially where full-width punctuation marks are used. This check will take that into account.

# purepunc (\*args, \*\*kwargs)

Checks that strings that are purely punctuation are not changed.

This extracts strings like + or - as these usually should not be changed.

## pythonbraceformat (\*args, \*\*kwargs)

Checks whether python brace format strings match.

#### run filters (unit, categorised=False)

Do some optimisation by caching some data of the unit for the benefit of run\_test().

### run\_test (test, unit)

Runs the given test on the given unit.

Note that this can raise a *FilterFailure* as part of normal operation.

# sentencecount (\*args, \*\*kwargs)

Checks that the number of sentences in both strings match.

Adds the number of sentences to see that the sentence count is the same between the original and translated string. You may not always want to use this test, if you find you often need to reformat your translation, because the original is badly-expressed, or because the structure of your language works better that way. Do what works best for your language: it's the meaning of the original you want to convey, not the exact way it was written in the English.

### setconfig(config)

Sets the accelerator list.

#### setsuggestionstore(store)

Sets the filename that a checker should use for evaluating suggestions.

```
short (*args, **kwargs)
```

Checks whether a translation is much shorter than the original string.

This is most useful in the special case where the translation is 1 characters long while the source text is multiple characters long. Otherwise, we use a general ratio that will catch very big differences but is set conservatively to limit the number of false positives.

```
simplecaps (*args, **kwargs)
```

Checks the capitalisation of two strings isn't wildly different.

This will pick up many false positives, so don't be a slave to it. It is useful for identifying translations that don't start with a capital letter (upper-case letter) when they should, or those that do when they shouldn't. It will also highlight sentences that have extra capitals; depending on the capitalisation convention of your language, you might want to change these to Title Case, or change them all to normal sentence case.

## simpleplurals(\*args, \*\*kwargs)

Checks for English style plural(s) for you to review.

This test will extract any message that contains words with a final "(s)" in the source text. You can then inspect the message, to check that the correct plural form has been used for your language. In some languages, plurals are made by adding text at the beginning of words, making the English style messy. In this case, they often revert to the plural form. This test allows an editor to check that the plurals used are correct. Be aware that this test may create a number of false positives.

For languages with no plural forms (only one noun form) this test will simply test that nothing like "(s)" was used in the translation.

#### singlequoting(\*args, \*\*kwargs)

Checks whether singlequoting is consistent between the two strings.

The same as doublequoting but checks for the 'character. Because this is used in contractions like it's and in possessive forms like user's, this test can output spurious errors if your language doesn't use such forms. If a quote appears at the end of a sentence in the translation, i.e. '., this might not be detected properly by the check.

### spellcheck (\*args, \*\*kwargs)

Checks words that don't pass a spell check.

This test will check for misspelled words in your translation. The test first checks for misspelled words in the original (usually English) text, and adds those to an exclusion list. The advantage of this exclusion is that many words that are specific to the application will not raise errors e.g. program names, brand names, function names.

The checker works with PyEnchant. You need to have PyEnchant installed as well as a dictionary for your language (for example, one of the Hunspell or aspell dictionaries). This test will only work if you have specified the --language option.

The pofilter error that is created, lists the misspelled word, plus suggestions returned from the spell checker. That makes it easy for you to identify the word and select a replacement.

## startcaps (\*args, \*\*kwargs)

Checks that the message starts with the correct capitalisation.

After stripping whitespace and common punctuation characters, it then checks to see that the first remaining character is correctly capitalised. So, if the sentence starts with an upper-case letter, and the translation does not, an error is produced.

This check is entirely disabled for many languages that don't make a distinction between upper and lower case. Contact us if this is not yet disabled for your language.

# startpunc(\*args, \*\*kwargs)

Checks whether punctuation at the beginning of the strings match.

Operates as endpunc but you will probably see fewer errors.

#### startwhitespace(\*args, \*\*kwargs)

Checks whether whitespace at the beginning of the strings matches.

As in endwhitespace but you will see fewer errors.

# tabs (\*args, \*\*kwargs)

Checks whether tabs are consistent between the two strings.

Counts the number of \t tab markers and reports an error if they differ.

#### unchanged(\*args, \*\*kwargs)

Checks whether a translation is basically identical to the original string.

This checks to see if the translation isn't just a copy of the English original. Sometimes, this is what you want, but other times you will detect words that should have been translated.

### untranslated(\*args, \*\*kwargs)

Checks whether a string has been translated at all.

This check is really only useful if you want to extract untranslated strings so that they can be translated independently of the main work.

## urls (\*args, \*\*kwargs)

Checks that URLs are not translated.

This checks only basic URLs (http, ftp, mailto etc.) not all URIs (e.g. afp, smb, file). Generally, you don't want to translate URLs, unless they are example URLs (http://your\_server.com/filename.html). If the URL is for configuration information, then you need to query the developers about placing configuration information in PO files. It shouldn't really be there, unless it is very clearly marked: such information should go into a configuration file.

## validchars (\*args, \*\*kwargs)

Checks that only characters specified as valid appear in the translation.

Often during character conversion to and from UTF-8 you get some strange characters appearing in your translation. This test presents a simple way to try and identify such errors.

This test will only run of you specify the --validcharsfile command line option. This file contains all the characters that are valid in your language. You must use UTF-8 encoding for the characters in the file.

If the test finds any characters not in your valid characters file then the test will print the character together with its Unicode value (e.g. 002B).

#### variables (\*args, \*\*kwargs)

Checks whether variables of various forms are consistent between the two strings.

This checks to make sure that variables that appear in the original also appear in the translation. It can handle variables from projects like KDE or OpenOffice. It does not at the moment cope with variables that use the reordering syntax of Gettext PO files.

# xmltags (\*args, \*\*kwargs)

Checks that XML/HTML tags have not been translated.

This check finds the number of tags in the source string and checks that the same number are in the translation. If the counts don't match then either the tag is missing or it was mistakenly translated by the translator, both of which are errors.

The check ignores tags or things that look like tags that cover the whole string e.g. <Error> but will produce false positives for things like An <Error> occurred as here Error should be translated. It also will allow translation of the *alt* attribute in e.g. <img src="bob.png" alt="Image description"> or similar translatable attributes in OpenOffice.org help files.

class translate.filters.checks.IOSChecker(\*\*kwargs)

## accelerators (\*args, \*\*kwargs)

Checks whether accelerators are consistent between the two strings.

This test is capable of checking the different type of accelerators that are used in different projects, like Mozilla or KDE. The test will pick up accelerators that are missing and ones that shouldn't be there.

See accelerators on the localization guide for a full description on accelerators.

#### acronyms (\*args, \*\*kwargs)

Checks that acronyms that appear are unchanged.

If an acronym appears in the original this test will check that it appears in the translation. Translating acronyms is a language decision but many languages leave them unchanged. In that case this test is useful for tracking down translations of the acronym and correcting them.

#### blank (\*args, \*\*kwargs)

Checks whether a translation is totally blank.

This will check to see if a translation has inadvertently been translated as blank i.e. as spaces. This is different from untranslated which is completely empty. This test is useful in that if something is translated as "" it will appear to most tools as if it is translated.

#### brackets (\*args, \*\*kwargs)

Checks that the number of brackets in both strings match.

If ([{ or }]) appear in the original this will check that the same number appear in the translation.

#### checker name

Extract checker name, for example 'mozilla' from MozillaChecker.

## compendiumconflicts(\*args, \*\*kwargs)

Checks for Gettext compendium conflicts (#-#-#-#).

When you use msgcat to create a PO compendium it will insert #-#-#-# into entries that are not consistent. If the compendium is used later in a message merge then these conflicts will appear in your translations. This test quickly extracts those for correction.

### credits(\*args, \*\*kwargs)

Checks for messages containing translation credits instead of normal translations.

Some projects have consistent ways of giving credit to translators by having a unit or two where translators can fill in their name and possibly their contact details. This test allows you to find these units easily to check that they are completed correctly and also disables other tests that might incorrectly get triggered for these units (such as urls, emails, etc.)

### doublequoting(\*args, \*\*kwargs)

Checks whether doublequoting is consistent between the two strings.

Checks on double quotes " to ensure that you have the same number in both the original and the translated string. This tests takes into account that several languages use different quoting characters, and will test for them instead.

## doublespacing(\*args, \*\*kwargs)

Checks for bad double-spaces by comparing to original.

This will identify if you have [space][space] in when you don't have it in the original or it appears in the original but not in your translation. Some of these are spurious and how you correct them depends on the conventions of your language.

### doublewords (\*args, \*\*kwargs)

Checks for repeated words in the translation.

Words that have been repeated in a translation will be highlighted with this test e.g. "the the", "a a". These are generally typos that need correcting. Some languages may have valid repeated words in their structure, in that case either ignore those instances or switch this test off.

## emails (\*args, \*\*kwargs)

Checks that emails are not translated.

Generally you should not be translating email addresses. This check will look to see that email addresses e.g. info@example.com are not translated. In some cases of course you should translate the address but generally you shouldn't.

### endpunc (\*args, \*\*kwargs)

Checks whether punctuation at the end of the strings match.

This will ensure that the ending of your translation has the same punctuation as the original. E.g. if it ends in :[space] then so should yours. It is useful for ensuring that you have ellipses [...] in all your translations, not simply three separate full-stops. You may pick up some errors in the original: feel free to keep your translation and notify the programmers. In some languages, characters such as ? or ! are always preceded by a space e.g. [space]? — do what your language customs dictate. Other false positives you will notice are, for example, if through changes in word-order you add "), etc. at the end of the sentence. Do not change these: your language word-order takes precedence.

It must be noted that if you are tempted to leave out [full-stop] or [colon] or add [full-stop] to a sentence, that often these have been done for a reason, e.g. a list where fullstops make it look cluttered. So, initially match them with the English, and make changes once the program is being used.

This check is aware of several language conventions for punctuation characters, such as the custom question marks for Greek and Arabic, Devanagari Danda, full-width punctuation for CJK languages, etc. Support for your language can be added easily if it is not there yet.

# endwhitespace(\*args, \*\*kwargs)

Checks whether whitespace at the end of the strings matches.

Operates the same as endpunc but is only concerned with whitespace. This filter is particularly useful for those strings which will evidently be followed by another string in the program, e.g. [Password: ] or [Enter your username: ]. The whitespace is an inherent part of the string. This filter makes sure you don't miss those important but otherwise invisible spaces!

If your language uses full-width punctuation (like Chinese), the visual spacing in the character might be enough without an added extra space.

# escapes (\*args, \*\*kwargs)

Checks whether escaping is consistent between the two strings.

Checks escapes such as  $\n \u0000$  to ensure that if they exist in the original string you also have them in the translation.

## filepaths (\*args, \*\*kwargs)

Checks that file paths have not been translated.

Checks that paths such as /home/user1 have not been translated. Generally you do not translate a file path, unless it is being used as an example, e.g. your\_user\_name/path/to/filename.conf.

#### filteraccelerators by list(strl, acceptlist=None)

Filter out accelerators from str1.

## functions (\*args, \*\*kwargs)

Checks that function names are not translated.

Checks that function names e.g. rgb() or getEntity.Name() are not translated.

#### get\_ignored\_filters()

Return checker's additional filters for current language.

## getfilters (excludefilters=None, limitfilters=None)

Returns dictionary of available filters, including/excluding those in the given lists.

#### kdecomments (\*args, \*\*kwargs)

Checks to ensure that no KDE style comments appear in the translation.

KDE style translator comments appear in PO files as "\_: comment\n". New translators often translate the comment. This test tries to identify instances where the comment has been translated.

#### long (\*args, \*\*kwargs)

Checks whether a translation is much longer than the original string.

This is most useful in the special case where the translation is multiple characters long while the source text is only 1 character long. Otherwise, we use a general ratio that will catch very big differences but is set conservatively to limit the number of false positives.

# musttranslatewords(\*args, \*\*kwargs)

Checks that words configured as definitely translatable don't appear in the translation.

If for instance in your language you decide that you must translate 'OK' then this test will flag any occurrences of 'OK' in the translation if it appeared in the source string. You must specify a file containing all of the *must translate* words using --musttranslatefile.

## newlines (\*args, \*\*kwargs)

Checks whether newlines are consistent between the two strings.

Counts the number of \n newlines (and variants such as \r\n) and reports and error if they differ.

#### notranslatewords (\*args, \*\*kwargs)

Checks that words configured as untranslatable appear in the translation too.

Many brand names should not be translated, this test allows you to easily make sure that words like: Word, Excel, Impress, Calc, etc. are not translated. You must specify a file containing all of the *no translate* words using --notranslatefile.

#### numbers (\*args, \*\*kwargs)

Checks whether numbers of various forms are consistent between the two strings.

You will see some errors where you have either written the number in full or converted it to the digit in your translation. Also changes in order will trigger this error.

## options (\*args, \*\*kwargs)

Checks that command line options are not translated.

In messages that contain command line options, such as <code>--help</code>, this test will check that these remain untranslated. These could be translated in the future if programs can create a mechanism to allow this, but currently they are not translated. If the options has a parameter, e.g. <code>--file=FILE</code>, then the test will check that the parameter has been translated.

# printf(\*args, \*\*kwargs)

Checks whether printf format strings match.

If the printf formatting variables are not identical, then this will indicate an error. Printf statements are used by programs to format output in a human readable form (they are placeholders for variable data). They allow you to specify lengths of string variables, string padding, number padding, precision, etc. Generally they will look like this: %d, %5.2f, %100s, etc. The test can also manage variables-reordering using the %1\$s syntax. The variables' type and details following data are tested to ensure that they are strictly identical, but they may be reordered.

See also printf Format String.

#### puncspacing (\*args, \*\*kwargs)

Checks for bad spacing after punctuation.

In the case of [full-stop][space] in the original, this test checks that your translation does not remove the space. It checks also for [comma], [colon], etc.

Some languages don't use spaces after common punctuation marks, especially where full-width punctuation marks are used. This check will take that into account.

#### purepunc (\*args, \*\*kwargs)

Checks that strings that are purely punctuation are not changed.

This extracts strings like + or - as these usually should not be changed.

### pythonbraceformat (\*args, \*\*kwargs)

Checks whether python brace format strings match.

#### run\_filters (unit, categorised=False)

Do some optimisation by caching some data of the unit for the benefit of run test ().

#### run\_test (test, unit)

Runs the given test on the given unit.

Note that this can raise a *FilterFailure* as part of normal operation.

#### sentencecount (\*args, \*\*kwargs)

Checks that the number of sentences in both strings match.

Adds the number of sentences to see that the sentence count is the same between the original and translated string. You may not always want to use this test, if you find you often need to reformat your translation, because the original is badly-expressed, or because the structure of your language works better that way. Do what works best for your language: it's the meaning of the original you want to convey, not the exact way it was written in the English.

#### setconfig(config)

Sets the accelerator list.

#### setsuggestionstore(store)

Sets the filename that a checker should use for evaluating suggestions.

```
short (*args, **kwargs)
```

Checks whether a translation is much shorter than the original string.

This is most useful in the special case where the translation is 1 characters long while the source text is multiple characters long. Otherwise, we use a general ratio that will catch very big differences but is set conservatively to limit the number of false positives.

```
simplecaps (*args, **kwargs)
```

Checks the capitalisation of two strings isn't wildly different.

This will pick up many false positives, so don't be a slave to it. It is useful for identifying translations that don't start with a capital letter (upper-case letter) when they should, or those that do when they shouldn't. It will also highlight sentences that have extra capitals; depending on the capitalisation convention of your language, you might want to change these to Title Case, or change them all to normal sentence case.

### simpleplurals(\*args, \*\*kwargs)

Checks for English style plural(s) for you to review.

This test will extract any message that contains words with a final "(s)" in the source text. You can then inspect the message, to check that the correct plural form has been used for your language. In some languages, plurals are made by adding text at the beginning of words, making the English style messy. In this case, they often revert to the plural form. This test allows an editor to check that the plurals used are correct. Be aware that this test may create a number of false positives.

For languages with no plural forms (only one noun form) this test will simply test that nothing like "(s)" was used in the translation.

## singlequoting(\*args, \*\*kwargs)

Checks whether singlequoting is consistent between the two strings.

The same as doublequoting but checks for the 'character. Because this is used in contractions like it's and in possessive forms like user's, this test can output spurious errors if your language doesn't use such

forms. If a quote appears at the end of a sentence in the translation, i.e. '., this might not be detected properly by the check.

### spellcheck (\*args, \*\*kwargs)

Checks words that don't pass a spell check.

This test will check for misspelled words in your translation. The test first checks for misspelled words in the original (usually English) text, and adds those to an exclusion list. The advantage of this exclusion is that many words that are specific to the application will not raise errors e.g. program names, brand names, function names.

The checker works with PyEnchant. You need to have PyEnchant installed as well as a dictionary for your language (for example, one of the Hunspell or aspell dictionaries). This test will only work if you have specified the --language option.

The pofilter error that is created, lists the misspelled word, plus suggestions returned from the spell checker. That makes it easy for you to identify the word and select a replacement.

# startcaps (\*args, \*\*kwargs)

Checks that the message starts with the correct capitalisation.

After stripping whitespace and common punctuation characters, it then checks to see that the first remaining character is correctly capitalised. So, if the sentence starts with an upper-case letter, and the translation does not, an error is produced.

This check is entirely disabled for many languages that don't make a distinction between upper and lower case. Contact us if this is not yet disabled for your language.

## startpunc(\*args, \*\*kwargs)

Checks whether punctuation at the beginning of the strings match.

Operates as endpunc but you will probably see fewer errors.

# startwhitespace(\*args, \*\*kwargs)

Checks whether whitespace at the beginning of the strings matches.

As in endwhitespace but you will see fewer errors.

```
tabs (*args, **kwargs)
```

Checks whether tabs are consistent between the two strings.

Counts the number of \t tab markers and reports an error if they differ.

### unchanged (\*args, \*\*kwargs)

Checks whether a translation is basically identical to the original string.

This checks to see if the translation isn't just a copy of the English original. Sometimes, this is what you want, but other times you will detect words that should have been translated.

#### untranslated(\*args, \*\*kwargs)

Checks whether a string has been translated at all.

This check is really only useful if you want to extract untranslated strings so that they can be translated independently of the main work.

### urls (\*args, \*\*kwargs)

Checks that URLs are not translated.

This checks only basic URLs (http, ftp, mailto etc.) not all URIs (e.g. afp, smb, file). Generally, you don't want to translate URLs, unless they are example URLs (http://your\_server.com/filename.html). If the URL is for configuration information, then you need to query the developers about placing configuration information in PO files. It shouldn't really be there, unless it is very clearly marked: such information should go into a configuration file.

#### validchars (\*args, \*\*kwargs)

Checks that only characters specified as valid appear in the translation.

Often during character conversion to and from UTF-8 you get some strange characters appearing in your translation. This test presents a simple way to try and identify such errors.

This test will only run of you specify the --validcharsfile command line option. This file contains all the characters that are valid in your language. You must use UTF-8 encoding for the characters in the file.

If the test finds any characters not in your valid characters file then the test will print the character together with its Unicode value (e.g. 002B).

## variables (\*args, \*\*kwargs)

Checks whether variables of various forms are consistent between the two strings.

This checks to make sure that variables that appear in the original also appear in the translation. It can handle variables from projects like KDE or OpenOffice. It does not at the moment cope with variables that use the reordering syntax of Gettext PO files.

# xmltags (\*args, \*\*kwargs)

Checks that XML/HTML tags have not been translated.

This check finds the number of tags in the source string and checks that the same number are in the translation. If the counts don't match then either the tag is missing or it was mistakenly translated by the translator, both of which are errors.

The check ignores tags or things that look like tags that cover the whole string e.g. <Error> but will produce false positives for things like An <Error> occurred as here Error should be translated. It also will allow translation of the *alt* attribute in e.g. <img src="bob.png" alt="Image description"> or similar translatable attributes in OpenOffice.org help files.

class translate.filters.checks.KdeChecker(\*\*kwargs)

### accelerators (\*args, \*\*kwargs)

Checks whether accelerators are consistent between the two strings.

This test is capable of checking the different type of accelerators that are used in different projects, like Mozilla or KDE. The test will pick up accelerators that are missing and ones that shouldn't be there.

See accelerators on the localization guide for a full description on accelerators.

# acronyms (\*args, \*\*kwargs)

Checks that acronyms that appear are unchanged.

If an acronym appears in the original this test will check that it appears in the translation. Translating acronyms is a language decision but many languages leave them unchanged. In that case this test is useful for tracking down translations of the acronym and correcting them.

## blank (\*args, \*\*kwargs)

Checks whether a translation is totally blank.

This will check to see if a translation has inadvertently been translated as blank i.e. as spaces. This is different from untranslated which is completely empty. This test is useful in that if something is translated as "" it will appear to most tools as if it is translated.

## brackets (\*args, \*\*kwargs)

Checks that the number of brackets in both strings match.

If ([{ or }]) appear in the original this will check that the same number appear in the translation.

#### checker name

Extract checker name, for example 'mozilla' from MozillaChecker.

#### compendiumconflicts(\*args, \*\*kwargs)

Checks for Gettext compendium conflicts (#-#-#-#).

When you use msgcat to create a PO compendium it will insert #-#-#-# into entries that are not consistent. If the compendium is used later in a message merge then these conflicts will appear in your translations. This test quickly extracts those for correction.

# credits(\*args, \*\*kwargs)

Checks for messages containing translation credits instead of normal translations.

Some projects have consistent ways of giving credit to translators by having a unit or two where translators can fill in their name and possibly their contact details. This test allows you to find these units easily to check that they are completed correctly and also disables other tests that might incorrectly get triggered for these units (such as urls, emails, etc.)

## doublequoting(\*args, \*\*kwargs)

Checks whether doublequoting is consistent between the two strings.

Checks on double quotes " to ensure that you have the same number in both the original and the translated string. This tests takes into account that several languages use different quoting characters, and will test for them instead.

## doublespacing(\*args, \*\*kwargs)

Checks for bad double-spaces by comparing to original.

This will identify if you have [space] [space] in when you don't have it in the original or it appears in the original but not in your translation. Some of these are spurious and how you correct them depends on the conventions of your language.

# doublewords (\*args, \*\*kwargs)

Checks for repeated words in the translation.

Words that have been repeated in a translation will be highlighted with this test e.g. "the the", "a a". These are generally typos that need correcting. Some languages may have valid repeated words in their structure, in that case either ignore those instances or switch this test off.

### emails (\*args, \*\*kwargs)

Checks that emails are not translated.

Generally you should not be translating email addresses. This check will look to see that email addresses e.g. info@example.com are not translated. In some cases of course you should translate the address but generally you shouldn't.

#### endpunc (\*args, \*\*kwargs)

Checks whether punctuation at the end of the strings match.

This will ensure that the ending of your translation has the same punctuation as the original. E.g. if it ends in :[space] then so should yours. It is useful for ensuring that you have ellipses [...] in all your translations, not simply three separate full-stops. You may pick up some errors in the original: feel free to keep your translation and notify the programmers. In some languages, characters such as ? or ! are always preceded by a space e.g. [space]? — do what your language customs dictate. Other false positives you will notice are, for example, if through changes in word-order you add "), etc. at the end of the sentence. Do not change these: your language word-order takes precedence.

It must be noted that if you are tempted to leave out [full-stop] or [colon] or add [full-stop] to a sentence, that often these have been done for a reason, e.g. a list where fullstops make it look cluttered. So, initially match them with the English, and make changes once the program is being used.

This check is aware of several language conventions for punctuation characters, such as the custom question marks for Greek and Arabic, Devanagari Danda, full-width punctuation for CJK languages, etc. Support for your language can be added easily if it is not there yet.

### endwhitespace(\*args, \*\*kwargs)

Checks whether whitespace at the end of the strings matches.

Operates the same as endpunc but is only concerned with whitespace. This filter is particularly useful for those strings which will evidently be followed by another string in the program, e.g. [Password: ] or [Enter your username: ]. The whitespace is an inherent part of the string. This filter makes sure you don't miss those important but otherwise invisible spaces!

If your language uses full-width punctuation (like Chinese), the visual spacing in the character might be enough without an added extra space.

#### escapes (\*args, \*\*kwargs)

Checks whether escaping is consistent between the two strings.

Checks escapes such as  $\n \u0000$  to ensure that if they exist in the original string you also have them in the translation.

### filepaths (\*args, \*\*kwargs)

Checks that file paths have not been translated.

Checks that paths such as /home/user1 have not been translated. Generally you do not translate a file path, unless it is being used as an example, e.g. your\_user\_name/path/to/filename.conf.

## filteraccelerators\_by\_list(str1, acceptlist=None)

Filter out accelerators from str1.

### functions (\*args, \*\*kwargs)

Checks that function names are not translated.

Checks that function names e.g. rgb() or getEntity.Name() are not translated.

## get\_ignored\_filters()

Return checker's additional filters for current language.

#### getfilters (excludefilters=None, limitfilters=None)

Returns dictionary of available filters, including/excluding those in the given lists.

## kdecomments (\*args, \*\*kwargs)

Checks to ensure that no KDE style comments appear in the translation.

KDE style translator comments appear in PO files as "\_: comment\n". New translators often translate the comment. This test tries to identify instances where the comment has been translated.

#### long(\*args, \*\*kwargs)

Checks whether a translation is much longer than the original string.

This is most useful in the special case where the translation is multiple characters long while the source text is only 1 character long. Otherwise, we use a general ratio that will catch very big differences but is set conservatively to limit the number of false positives.

#### musttranslatewords (\*args, \*\*kwargs)

Checks that words configured as definitely translatable don't appear in the translation.

If for instance in your language you decide that you must translate 'OK' then this test will flag any occurrences of 'OK' in the translation if it appeared in the source string. You must specify a file containing all of the *must translate* words using --musttranslatefile.

#### newlines (\*args, \*\*kwargs)

Checks whether newlines are consistent between the two strings.

Counts the number of \n newlines (and variants such as \r\n) and reports and error if they differ.

#### notranslatewords (\*args, \*\*kwargs)

Checks that words configured as untranslatable appear in the translation too.

Many brand names should not be translated, this test allows you to easily make sure that words like: Word, Excel, Impress, Calc, etc. are not translated. You must specify a file containing all of the *no translate* words using --notranslatefile.

```
numbers (*args, **kwargs)
```

Checks whether numbers of various forms are consistent between the two strings.

You will see some errors where you have either written the number in full or converted it to the digit in your translation. Also changes in order will trigger this error.

```
options (*args, **kwargs)
```

Checks that command line options are not translated.

In messages that contain command line options, such as <code>--help</code>, this test will check that these remain untranslated. These could be translated in the future if programs can create a mechanism to allow this, but currently they are not translated. If the options has a parameter, e.g. <code>--file=FILE</code>, then the test will check that the parameter has been translated.

```
printf(*args, **kwargs)
```

Checks whether printf format strings match.

If the printf formatting variables are not identical, then this will indicate an error. Printf statements are used by programs to format output in a human readable form (they are placeholders for variable data). They allow you to specify lengths of string variables, string padding, number padding, precision, etc. Generally they will look like this: %d, %5.2f, %100s, etc. The test can also manage variables-reordering using the %1\$s syntax. The variables' type and details following data are tested to ensure that they are strictly identical, but they may be reordered.

See also printf Format String.

```
puncspacing(*args, **kwargs)
```

Checks for bad spacing after punctuation.

In the case of [full-stop][space] in the original, this test checks that your translation does not remove the space. It checks also for [comma], [colon], etc.

Some languages don't use spaces after common punctuation marks, especially where full-width punctuation marks are used. This check will take that into account.

```
purepunc (*args, **kwargs)
```

Checks that strings that are purely punctuation are not changed.

This extracts strings like + or - as these usually should not be changed.

```
pythonbraceformat (*args, **kwargs)
```

Checks whether python brace format strings match.

## run\_filters (unit, categorised=False)

Do some optimisation by caching some data of the unit for the benefit of run\_test().

#### run\_test (test, unit)

Runs the given test on the given unit.

Note that this can raise a FilterFailure as part of normal operation.

## sentencecount (\*args, \*\*kwargs)

Checks that the number of sentences in both strings match.

Adds the number of sentences to see that the sentence count is the same between the original and translated string. You may not always want to use this test, if you find you often need to reformat your translation, because the original is badly-expressed, or because the structure of your language works better that way. Do what works best for your language: it's the meaning of the original you want to convey, not the exact way it was written in the English.

#### setconfig(config)

Sets the accelerator list.

#### setsuggestionstore(store)

Sets the filename that a checker should use for evaluating suggestions.

## short (\*args, \*\*kwargs)

Checks whether a translation is much shorter than the original string.

This is most useful in the special case where the translation is 1 characters long while the source text is multiple characters long. Otherwise, we use a general ratio that will catch very big differences but is set conservatively to limit the number of false positives.

### simplecaps (\*args, \*\*kwargs)

Checks the capitalisation of two strings isn't wildly different.

This will pick up many false positives, so don't be a slave to it. It is useful for identifying translations that don't start with a capital letter (upper-case letter) when they should, or those that do when they shouldn't. It will also highlight sentences that have extra capitals; depending on the capitalisation convention of your language, you might want to change these to Title Case, or change them all to normal sentence case.

## simpleplurals(\*args, \*\*kwargs)

Checks for English style plural(s) for you to review.

This test will extract any message that contains words with a final "(s)" in the source text. You can then inspect the message, to check that the correct plural form has been used for your language. In some languages, plurals are made by adding text at the beginning of words, making the English style messy. In this case, they often revert to the plural form. This test allows an editor to check that the plurals used are correct. Be aware that this test may create a number of false positives.

For languages with no plural forms (only one noun form) this test will simply test that nothing like "(s)" was used in the translation.

#### singlequoting(\*args, \*\*kwargs)

Checks whether singlequoting is consistent between the two strings.

The same as doublequoting but checks for the 'character. Because this is used in contractions like it's and in possessive forms like user's, this test can output spurious errors if your language doesn't use such forms. If a quote appears at the end of a sentence in the translation, i.e. '., this might not be detected properly by the check.

#### spellcheck (\*args, \*\*kwargs)

Checks words that don't pass a spell check.

This test will check for misspelled words in your translation. The test first checks for misspelled words in the original (usually English) text, and adds those to an exclusion list. The advantage of this exclusion is that many words that are specific to the application will not raise errors e.g. program names, brand names, function names.

The checker works with PyEnchant. You need to have PyEnchant installed as well as a dictionary for your language (for example, one of the Hunspell or aspell dictionaries). This test will only work if you have specified the --language option.

The pofilter error that is created, lists the misspelled word, plus suggestions returned from the spell checker. That makes it easy for you to identify the word and select a replacement.

#### startcaps (\*args, \*\*kwargs)

Checks that the message starts with the correct capitalisation.

After stripping whitespace and common punctuation characters, it then checks to see that the first remaining character is correctly capitalised. So, if the sentence starts with an upper-case letter, and the translation does not, an error is produced.

This check is entirely disabled for many languages that don't make a distinction between upper and lower case. Contact us if this is not yet disabled for your language.

## startpunc(\*args, \*\*kwargs)

Checks whether punctuation at the beginning of the strings match.

Operates as endpunc but you will probably see fewer errors.

# startwhitespace(\*args, \*\*kwargs)

Checks whether whitespace at the beginning of the strings matches.

As in endwhitespace but you will see fewer errors.

## tabs (\*args, \*\*kwargs)

Checks whether tabs are consistent between the two strings.

Counts the number of \t tab markers and reports an error if they differ.

# unchanged(\*args, \*\*kwargs)

Checks whether a translation is basically identical to the original string.

This checks to see if the translation isn't just a copy of the English original. Sometimes, this is what you want, but other times you will detect words that should have been translated.

## untranslated(\*args, \*\*kwargs)

Checks whether a string has been translated at all.

This check is really only useful if you want to extract untranslated strings so that they can be translated independently of the main work.

#### urls (\*args, \*\*kwargs)

Checks that URLs are not translated.

This checks only basic URLs (http, ftp, mailto etc.) not all URIs (e.g. afp, smb, file). Generally, you don't want to translate URLs, unless they are example URLs (http://your\_server.com/filename.html). If the URL is for configuration information, then you need to query the developers about placing configuration information in PO files. It shouldn't really be there, unless it is very clearly marked: such information should go into a configuration file.

# validchars (\*args, \*\*kwargs)

Checks that only characters specified as valid appear in the translation.

Often during character conversion to and from UTF-8 you get some strange characters appearing in your translation. This test presents a simple way to try and identify such errors.

This test will only run of you specify the --validcharsfile command line option. This file contains all the characters that are valid in your language. You must use UTF-8 encoding for the characters in the file.

If the test finds any characters not in your valid characters file then the test will print the character together with its Unicode value (e.g. 002B).

## variables (\*args, \*\*kwargs)

Checks whether variables of various forms are consistent between the two strings.

This checks to make sure that variables that appear in the original also appear in the translation. It can handle variables from projects like KDE or OpenOffice. It does not at the moment cope with variables that use the reordering syntax of Gettext PO files.

# xmltags (\*args, \*\*kwargs)

Checks that XML/HTML tags have not been translated.

This check finds the number of tags in the source string and checks that the same number are in the translation. If the counts don't match then either the tag is missing or it was mistakenly translated by the translator, both of which are errors.

The check ignores tags or things that look like tags that cover the whole string e.g. <Error> but will produce false positives for things like An <Error> occurred as here Error should be translated. It also will allow translation of the *alt* attribute in e.g. <img src="bob.png" alt="Image description"> or similar translatable attributes in OpenOffice.org help files.

class translate.filters.checks.L20nChecker(\*\*kwargs)

### accelerators (\*args, \*\*kwargs)

Checks whether accelerators are consistent between the two strings.

For Mozilla we lower the severity to cosmetic, and for some languages it also ensures accelerators are absent in the target string since some languages do not use accelerators, for example Indic languages.

```
acronyms (*args, **kwargs)
```

Checks that acronyms that appear are unchanged.

If an acronym appears in the original this test will check that it appears in the translation. Translating acronyms is a language decision but many languages leave them unchanged. In that case this test is useful for tracking down translations of the acronym and correcting them.

```
blank (*args, **kwargs)
```

Checks whether a translation is totally blank.

This will check to see if a translation has inadvertently been translated as blank i.e. as spaces. This is different from untranslated which is completely empty. This test is useful in that if something is translated as "" it will appear to most tools as if it is translated.

```
brackets (*args, **kwargs)
```

Checks that the number of brackets in both strings match.

If ( [ { or } ] ) appear in the original this will check that the same number appear in the translation.

#### checker name

Extract checker name, for example 'mozilla' from MozillaChecker.

#### compendiumconflicts(\*args, \*\*kwargs)

Checks for Gettext compendium conflicts (#-#-#-#).

When you use msgcat to create a PO compendium it will insert #-#-#-#-# into entries that are not consistent. If the compendium is used later in a message merge then these conflicts will appear in your translations. This test quickly extracts those for correction.

```
credits(*args, **kwargs)
```

Checks for messages containing translation credits instead of normal translations.

Some projects have consistent ways of giving credit to translators by having a unit or two where translators can fill in their name and possibly their contact details. This test allows you to find these units easily to check that they are completed correctly and also disables other tests that might incorrectly get triggered for these units (such as urls, emails, etc.)

#### dialogsizes (\*args, \*\*kwargs)

Checks that dialog sizes are not translated.

This is a Mozilla specific test. Mozilla uses a language called XUL to define dialogues and screens. This can make use of CSS to specify properties of the dialogue. These properties include things such as the width and height of the box. The size might need to be changed if the dialogue size changes due to longer translations. Thus translators can change these settings. But you are only meant to change the number not translate the words 'width' or 'height'. This check capture instances where these are translated. It will also catch other types of errors in these units.

## doublequoting(\*args, \*\*kwargs)

Checks whether doublequoting is consistent between the two strings.

Checks on double quotes " to ensure that you have the same number in both the original and the translated string. This tests takes into account that several languages use different quoting characters, and will test for them instead.

## doublespacing(\*args, \*\*kwargs)

Checks for bad double-spaces by comparing to original.

This will identify if you have [space][space] in when you don't have it in the original or it appears in the original but not in your translation. Some of these are spurious and how you correct them depends on the conventions of your language.

#### doublewords (\*args, \*\*kwargs)

Checks for repeated words in the translation.

Words that have been repeated in a translation will be highlighted with this test e.g. "the the", "a a". These are generally typos that need correcting. Some languages may have valid repeated words in their structure, in that case either ignore those instances or switch this test off.

## emails (\*args, \*\*kwargs)

Checks that emails are not translated.

Generally you should not be translating email addresses. This check will look to see that email addresses e.g. info@example.com are not translated. In some cases of course you should translate the address but generally you shouldn't.

## endpunc (\*args, \*\*kwargs)

Checks whether punctuation at the end of the strings match.

This will ensure that the ending of your translation has the same punctuation as the original. E.g. if it ends in :[space] then so should yours. It is useful for ensuring that you have ellipses [...] in all your translations, not simply three separate full-stops. You may pick up some errors in the original: feel free to keep your translation and notify the programmers. In some languages, characters such as ? or ! are always preceded by a space e.g. [space]? — do what your language customs dictate. Other false positives you will notice are, for example, if through changes in word-order you add "), etc. at the end of the sentence. Do not change these: your language word-order takes precedence.

It must be noted that if you are tempted to leave out [full-stop] or [colon] or add [full-stop] to a sentence, that often these have been done for a reason, e.g. a list where fullstops make it look cluttered. So, initially match them with the English, and make changes once the program is being used.

This check is aware of several language conventions for punctuation characters, such as the custom question marks for Greek and Arabic, Devanagari Danda, full-width punctuation for CJK languages, etc. Support for your language can be added easily if it is not there yet.

## endwhitespace(\*args, \*\*kwargs)

Checks whether whitespace at the end of the strings matches.

Operates the same as endpunc but is only concerned with whitespace. This filter is particularly useful for those strings which will evidently be followed by another string in the program, e.g. [Password: ] or [Enter your username: ]. The whitespace is an inherent part of the string. This filter makes sure you don't miss those important but otherwise invisible spaces!

If your language uses full-width punctuation (like Chinese), the visual spacing in the character might be enough without an added extra space.

#### escapes (\*args, \*\*kwargs)

Checks whether escaping is consistent between the two strings.

Checks escapes such as \n \u0000 to ensure that if they exist in the original string you also have them in the translation.

## filepaths (\*args, \*\*kwargs)

Checks that file paths have not been translated.

Checks that paths such as /home/user1 have not been translated. Generally you do not translate a file path, unless it is being used as an example, e.g. your\_user\_name/path/to/filename.conf.

#### filteraccelerators by list(strl, acceptlist=None)

Filter out accelerators from str1.

## ftl\_format (\*args, \*\*kwargs)

Checks for valid ftl syntax.

This checks the translation to detect any ftl parsing and syntax errors.

### functions (\*args, \*\*kwargs)

Checks that function names are not translated.

Checks that function names e.g. rgb() or getEntity.Name() are not translated.

### get\_ignored\_filters()

Return checker's additional filters for current language.

### getfilters (excludefilters=None, limitfilters=None)

Returns dictionary of available filters, including/excluding those in the given lists.

### kdecomments (\*args, \*\*kwargs)

Checks to ensure that no KDE style comments appear in the translation.

KDE style translator comments appear in PO files as "\_: comment\n". New translators often translate the comment. This test tries to identify instances where the comment has been translated.

## long(\*args, \*\*kwargs)

Checks whether a translation is much longer than the original string.

This is most useful in the special case where the translation is multiple characters long while the source text is only 1 character long. Otherwise, we use a general ratio that will catch very big differences but is set conservatively to limit the number of false positives.

## musttranslatewords(\*args, \*\*kwargs)

Checks that words configured as definitely translatable don't appear in the translation.

If for instance in your language you decide that you must translate 'OK' then this test will flag any occurrences of 'OK' in the translation if it appeared in the source string. You must specify a file containing all of the *must translate* words using --musttranslatefile.

## newlines(\*args, \*\*kwargs)

Checks whether newlines are consistent between the two strings.

Counts the number of  $\n$  newlines (and variants such as  $\n$ ) and reports and error if they differ.

#### notranslatewords (\*args, \*\*kwargs)

Checks that words configured as untranslatable appear in the translation too.

Many brand names should not be translated, this test allows you to easily make sure that words like: Word, Excel, Impress, Calc, etc. are not translated. You must specify a file containing all of the *no translate* words using --notranslatefile.

```
numbers (*args, **kwargs)
```

Checks that numbers are not translated.

Special handling for Mozilla to ignore entries that are dialog sizes.

```
options (*args, **kwargs)
```

Checks that command line options are not translated.

In messages that contain command line options, such as <code>--help</code>, this test will check that these remain untranslated. These could be translated in the future if programs can create a mechanism to allow this, but currently they are not translated. If the options has a parameter, e.g. <code>--file=FILE</code>, then the test will check that the parameter has been translated.

```
printf(*args, **kwargs)
```

Checks whether printf format strings match.

If the printf formatting variables are not identical, then this will indicate an error. Printf statements are used by programs to format output in a human readable form (they are placeholders for variable data). They allow you to specify lengths of string variables, string padding, number padding, precision, etc. Generally they will look like this: %d, %5.2f, %100s, etc. The test can also manage variables-reordering using the %1\$s syntax. The variables' type and details following data are tested to ensure that they are strictly identical, but they may be reordered.

See also printf Format String.

## puncspacing(\*args, \*\*kwargs)

Checks for bad spacing after punctuation.

In the case of [full-stop][space] in the original, this test checks that your translation does not remove the space. It checks also for [comma], [colon], etc.

Some languages don't use spaces after common punctuation marks, especially where full-width punctuation marks are used. This check will take that into account.

```
purepunc (*args, **kwargs)
```

Checks that strings that are purely punctuation are not changed.

This extracts strings like + or - as these usually should not be changed.

```
pythonbraceformat (*args, **kwargs)
```

Checks whether python brace format strings match.

```
run_filters (unit, categorised=False)
```

Do some optimisation by caching some data of the unit for the benefit of run\_test().

```
run_test (test, unit)
```

Runs the given test on the given unit.

Note that this can raise a *FilterFailure* as part of normal operation.

### sentencecount (\*args, \*\*kwargs)

Checks that the number of sentences in both strings match.

Adds the number of sentences to see that the sentence count is the same between the original and translated string. You may not always want to use this test, if you find you often need to reformat your translation, because the original is badly-expressed, or because the structure of your language works better that way.

Do what works best for your language: it's the meaning of the original you want to convey, not the exact way it was written in the English.

## setconfig(config)

Sets the accelerator list.

## setsuggestionstore(store)

Sets the filename that a checker should use for evaluating suggestions.

#### short (\*args, \*\*kwargs)

Checks whether a translation is much shorter than the original string.

This is most useful in the special case where the translation is 1 characters long while the source text is multiple characters long. Otherwise, we use a general ratio that will catch very big differences but is set conservatively to limit the number of false positives.

#### simplecaps (\*args, \*\*kwargs)

Checks the capitalisation of two strings isn't wildly different.

This will pick up many false positives, so don't be a slave to it. It is useful for identifying translations that don't start with a capital letter (upper-case letter) when they should, or those that do when they shouldn't. It will also highlight sentences that have extra capitals; depending on the capitalisation convention of your language, you might want to change these to Title Case, or change them all to normal sentence case.

### simpleplurals(\*args, \*\*kwargs)

Checks for English style plural(s) for you to review.

This test will extract any message that contains words with a final "(s)" in the source text. You can then inspect the message, to check that the correct plural form has been used for your language. In some languages, plurals are made by adding text at the beginning of words, making the English style messy. In this case, they often revert to the plural form. This test allows an editor to check that the plurals used are correct. Be aware that this test may create a number of false positives.

For languages with no plural forms (only one noun form) this test will simply test that nothing like "(s)" was used in the translation.

## singlequoting(\*args, \*\*kwargs)

Checks whether singlequoting is consistent between the two strings.

The same as doublequoting but checks for the 'character. Because this is used in contractions like it's and in possessive forms like user's, this test can output spurious errors if your language doesn't use such forms. If a quote appears at the end of a sentence in the translation, i.e. '., this might not be detected properly by the check.

### spellcheck (\*args, \*\*kwargs)

Checks words that don't pass a spell check.

This test will check for misspelled words in your translation. The test first checks for misspelled words in the original (usually English) text, and adds those to an exclusion list. The advantage of this exclusion is that many words that are specific to the application will not raise errors e.g. program names, brand names, function names.

The checker works with PyEnchant. You need to have PyEnchant installed as well as a dictionary for your language (for example, one of the Hunspell or aspell dictionaries). This test will only work if you have specified the --language option.

The pofilter error that is created, lists the misspelled word, plus suggestions returned from the spell checker. That makes it easy for you to identify the word and select a replacement.

# startcaps (\*args, \*\*kwargs)

Checks that the message starts with the correct capitalisation.

After stripping whitespace and common punctuation characters, it then checks to see that the first remaining character is correctly capitalised. So, if the sentence starts with an upper-case letter, and the translation does not, an error is produced.

This check is entirely disabled for many languages that don't make a distinction between upper and lower case. Contact us if this is not yet disabled for your language.

#### startpunc(\*args, \*\*kwargs)

Checks whether punctuation at the beginning of the strings match.

Operates as endpunc but you will probably see fewer errors.

## startwhitespace (\*args, \*\*kwargs)

Checks whether whitespace at the beginning of the strings matches.

As in endwhitespace but you will see fewer errors.

### tabs (\*args, \*\*kwargs)

Checks whether tabs are consistent between the two strings.

Counts the number of \t tab markers and reports an error if they differ.

### unchanged(\*args, \*\*kwargs)

Checks whether a translation is basically identical to the original string.

Special handling for Mozilla to ignore entries that are dialog sizes.

### untranslated(\*args, \*\*kwargs)

Checks whether a string has been translated at all.

This check is really only useful if you want to extract untranslated strings so that they can be translated independently of the main work.

## urls(\*args, \*\*kwargs)

Checks that URLs are not translated.

This checks only basic URLs (http, ftp, mailto etc.) not all URIs (e.g. afp, smb, file). Generally, you don't want to translate URLs, unless they are example URLs (http://your\_server.com/filename.html). If the URL is for configuration information, then you need to query the developers about placing configuration information in PO files. It shouldn't really be there, unless it is very clearly marked: such information should go into a configuration file.

## validchars (\*args, \*\*kwargs)

Checks that only characters specified as valid appear in the translation.

Often during character conversion to and from UTF-8 you get some strange characters appearing in your translation. This test presents a simple way to try and identify such errors.

This test will only run of you specify the --validcharsfile command line option. This file contains all the characters that are valid in your language. You must use UTF-8 encoding for the characters in the file.

If the test finds any characters not in your valid characters file then the test will print the character together with its Unicode value (e.g. 002B).

### variables (\*args, \*\*kwargs)

Checks whether variables of various forms are consistent between the two strings.

This checks to make sure that variables that appear in the original also appear in the translation. It can handle variables from projects like KDE or OpenOffice. It does not at the moment cope with variables that use the reordering syntax of Gettext PO files.

#### xmltags (\*args, \*\*kwargs)

Checks that XML/HTML tags have not been translated.

This check finds the number of tags in the source string and checks that the same number are in the translation. If the counts don't match then either the tag is missing or it was mistakenly translated by the translator, both of which are errors.

The check ignores tags or things that look like tags that cover the whole string e.g. <Error> but will produce false positives for things like An <Error> occurred as here Error should be translated. It also will allow translation of the *alt* attribute in e.g. <img src="bob.png" alt="Image description"> or similar translatable attributes in OpenOffice.org help files.

class translate.filters.checks.LibreOfficeChecker(\*\*kwargs)

## accelerators (\*args, \*\*kwargs)

Checks whether accelerators are consistent between the two strings.

This test is capable of checking the different type of accelerators that are used in different projects, like Mozilla or KDE. The test will pick up accelerators that are missing and ones that shouldn't be there.

See accelerators on the localization guide for a full description on accelerators.

## acronyms (\*args, \*\*kwargs)

Checks that acronyms that appear are unchanged.

If an acronym appears in the original this test will check that it appears in the translation. Translating acronyms is a language decision but many languages leave them unchanged. In that case this test is useful for tracking down translations of the acronym and correcting them.

#### blank (\*args, \*\*kwargs)

Checks whether a translation is totally blank.

This will check to see if a translation has inadvertently been translated as blank i.e. as spaces. This is different from untranslated which is completely empty. This test is useful in that if something is translated as "" it will appear to most tools as if it is translated.

#### brackets (\*args, \*\*kwargs)

Checks that the number of brackets in both strings match.

If ([{ or }]) appear in the original this will check that the same number appear in the translation.

#### checker\_name

Extract checker name, for example 'mozilla' from MozillaChecker.

## compendiumconflicts(\*args, \*\*kwargs)

Checks for Gettext compendium conflicts (#-#-#-#).

When you use msgcat to create a PO compendium it will insert #-#-#-# into entries that are not consistent. If the compendium is used later in a message merge then these conflicts will appear in your translations. This test quickly extracts those for correction.

### credits (\*args, \*\*kwargs)

Checks for messages containing translation credits instead of normal translations.

Some projects have consistent ways of giving credit to translators by having a unit or two where translators can fill in their name and possibly their contact details. This test allows you to find these units easily to check that they are completed correctly and also disables other tests that might incorrectly get triggered for these units (such as urls, emails, etc.)

## doublequoting(\*args, \*\*kwargs)

Checks whether doublequoting is consistent between the two strings.

Checks on double quotes " to ensure that you have the same number in both the original and the translated string. This tests takes into account that several languages use different quoting characters, and will test for them instead.

#### doublespacing(\*args, \*\*kwargs)

Checks for bad double-spaces by comparing to original.

This will identify if you have [space][space] in when you don't have it in the original or it appears in the original but not in your translation. Some of these are spurious and how you correct them depends on the conventions of your language.

#### doublewords (\*args, \*\*kwargs)

Checks for repeated words in the translation.

Words that have been repeated in a translation will be highlighted with this test e.g. "the the", "a a". These are generally typos that need correcting. Some languages may have valid repeated words in their structure, in that case either ignore those instances or switch this test off.

## emails (\*args, \*\*kwargs)

Checks that emails are not translated.

Generally you should not be translating email addresses. This check will look to see that email addresses e.g. info@example.com are not translated. In some cases of course you should translate the address but generally you shouldn't.

### endpunc (\*args, \*\*kwargs)

Checks whether punctuation at the end of the strings match.

This will ensure that the ending of your translation has the same punctuation as the original. E.g. if it ends in :[space] then so should yours. It is useful for ensuring that you have ellipses [...] in all your translations, not simply three separate full-stops. You may pick up some errors in the original: feel free to keep your translation and notify the programmers. In some languages, characters such as ? or ! are always preceded by a space e.g. [space]? — do what your language customs dictate. Other false positives you will notice are, for example, if through changes in word-order you add "), etc. at the end of the sentence. Do not change these: your language word-order takes precedence.

It must be noted that if you are tempted to leave out [full-stop] or [colon] or add [full-stop] to a sentence, that often these have been done for a reason, e.g. a list where fullstops make it look cluttered. So, initially match them with the English, and make changes once the program is being used.

This check is aware of several language conventions for punctuation characters, such as the custom question marks for Greek and Arabic, Devanagari Danda, full-width punctuation for CJK languages, etc. Support for your language can be added easily if it is not there yet.

## endwhitespace(\*args, \*\*kwargs)

Checks whether whitespace at the end of the strings matches.

Operates the same as endpunc but is only concerned with whitespace. This filter is particularly useful for those strings which will evidently be followed by another string in the program, e.g. [Password: ] or [Enter your username: ]. The whitespace is an inherent part of the string. This filter makes sure you don't miss those important but otherwise invisible spaces!

If your language uses full-width punctuation (like Chinese), the visual spacing in the character might be enough without an added extra space.

### escapes (\*args, \*\*kwargs)

Checks whether escaping is consistent between the two strings.

Checks escapes such as  $\n \u0000$  to ensure that if they exist in the original string you also have them in the translation.

## filepaths (\*args, \*\*kwargs)

Checks that file paths have not been translated.

Checks that paths such as /home/user1 have not been translated. Generally you do not translate a file path, unless it is being used as an example, e.g. your user name/path/to/filename.conf.

#### filteraccelerators\_by\_list(str1, acceptlist=None)

Filter out accelerators from str1.

### functions (\*args, \*\*kwargs)

Checks that function names are not translated.

Checks that function names e.g. rgb() or getEntity.Name() are not translated.

#### get\_ignored\_filters()

Return checker's additional filters for current language.

#### getfilters (excludefilters=None, limitfilters=None)

Returns dictionary of available filters, including/excluding those in the given lists.

#### kdecomments (\*args, \*\*kwargs)

Checks to ensure that no KDE style comments appear in the translation.

KDE style translator comments appear in PO files as "\_: comment\n". New translators often translate the comment. This test tries to identify instances where the comment has been translated.

#### long (\*args, \*\*kwargs)

Checks whether a translation is much longer than the original string.

This is most useful in the special case where the translation is multiple characters long while the source text is only 1 character long. Otherwise, we use a general ratio that will catch very big differences but is set conservatively to limit the number of false positives.

#### musttranslatewords (\*args, \*\*kwargs)

Checks that words configured as definitely translatable don't appear in the translation.

If for instance in your language you decide that you must translate 'OK' then this test will flag any occurrences of 'OK' in the translation if it appeared in the source string. You must specify a file containing all of the *must translate* words using --musttranslatefile.

#### newlines (\*args, \*\*kwargs)

Checks whether newlines are consistent between the two strings.

Counts the number of \n newlines (and variants such as \r\n) and reports and error if they differ.

### notranslatewords (\*args, \*\*kwargs)

Checks that words configured as untranslatable appear in the translation too.

Many brand names should not be translated, this test allows you to easily make sure that words like: Word, Excel, Impress, Calc, etc. are not translated. You must specify a file containing all of the *no translate* words using --notranslatefile.

### numbers (\*args, \*\*kwargs)

Checks whether numbers of various forms are consistent between the two strings.

You will see some errors where you have either written the number in full or converted it to the digit in your translation. Also changes in order will trigger this error.

# options (\*args, \*\*kwargs)

Checks that command line options are not translated.

In messages that contain command line options, such as <code>--help</code>, this test will check that these remain untranslated. These could be translated in the future if programs can create a mechanism to allow this, but currently they are not translated. If the options has a parameter, e.g. <code>--file=FILE</code>, then the test will check that the parameter has been translated.

# printf(\*args, \*\*kwargs)

Checks whether printf format strings match.

If the printf formatting variables are not identical, then this will indicate an error. Printf statements are used by programs to format output in a human readable form (they are placeholders for variable data). They allow you to specify lengths of string variables, string padding, number padding, precision, etc. Generally they will look like this: %d, %5.2f, %100s, etc. The test can also manage variables-reordering using the %1\$s syntax. The variables' type and details following data are tested to ensure that they are strictly identical, but they may be reordered.

See also printf Format String.

### puncspacing(\*args, \*\*kwargs)

Checks for bad spacing after punctuation.

In the case of [full-stop][space] in the original, this test checks that your translation does not remove the space. It checks also for [comma], [colon], etc.

Some languages don't use spaces after common punctuation marks, especially where full-width punctuation marks are used. This check will take that into account.

## purepunc (\*args, \*\*kwargs)

Checks that strings that are purely punctuation are not changed.

This extracts strings like + or - as these usually should not be changed.

## pythonbraceformat (\*args, \*\*kwargs)

Not used in LibreOffice

#### run\_filters (unit, categorised=False)

Do some optimisation by caching some data of the unit for the benefit of run\_test().

#### run test(test, unit)

Runs the given test on the given unit.

Note that this can raise a FilterFailure as part of normal operation.

```
sentencecount (*args, **kwargs)
```

Checks that the number of sentences in both strings match.

Adds the number of sentences to see that the sentence count is the same between the original and translated string. You may not always want to use this test, if you find you often need to reformat your translation, because the original is badly-expressed, or because the structure of your language works better that way. Do what works best for your language: it's the meaning of the original you want to convey, not the exact way it was written in the English.

## setconfig(config)

Sets the accelerator list.

#### setsuggestionstore(store)

Sets the filename that a checker should use for evaluating suggestions.

```
short (*args, **kwargs)
```

Checks whether a translation is much shorter than the original string.

This is most useful in the special case where the translation is 1 characters long while the source text is multiple characters long. Otherwise, we use a general ratio that will catch very big differences but is set conservatively to limit the number of false positives.

#### simplecaps (\*args, \*\*kwargs)

Checks the capitalisation of two strings isn't wildly different.

This will pick up many false positives, so don't be a slave to it. It is useful for identifying translations that don't start with a capital letter (upper-case letter) when they should, or those that do when they shouldn't. It will also highlight sentences that have extra capitals; depending on the capitalisation convention of your language, you might want to change these to Title Case, or change them all to normal sentence case.

#### simpleplurals(\*args, \*\*kwargs)

Checks for English style plural(s) for you to review.

This test will extract any message that contains words with a final "(s)" in the source text. You can then inspect the message, to check that the correct plural form has been used for your language. In some languages, plurals are made by adding text at the beginning of words, making the English style messy. In this case, they often revert to the plural form. This test allows an editor to check that the plurals used are correct. Be aware that this test may create a number of false positives.

For languages with no plural forms (only one noun form) this test will simply test that nothing like "(s)" was used in the translation.

## singlequoting(\*args, \*\*kwargs)

Checks whether singlequoting is consistent between the two strings.

The same as doublequoting but checks for the 'character. Because this is used in contractions like it's and in possessive forms like user's, this test can output spurious errors if your language doesn't use such forms. If a quote appears at the end of a sentence in the translation, i.e. '., this might not be detected properly by the check.

## spellcheck (\*args, \*\*kwargs)

Checks words that don't pass a spell check.

This test will check for misspelled words in your translation. The test first checks for misspelled words in the original (usually English) text, and adds those to an exclusion list. The advantage of this exclusion is that many words that are specific to the application will not raise errors e.g. program names, brand names, function names.

The checker works with PyEnchant. You need to have PyEnchant installed as well as a dictionary for your language (for example, one of the Hunspell or aspell dictionaries). This test will only work if you have specified the --language option.

The pofilter error that is created, lists the misspelled word, plus suggestions returned from the spell checker. That makes it easy for you to identify the word and select a replacement.

## startcaps (\*args, \*\*kwargs)

Checks that the message starts with the correct capitalisation.

After stripping whitespace and common punctuation characters, it then checks to see that the first remaining character is correctly capitalised. So, if the sentence starts with an upper-case letter, and the translation does not, an error is produced.

This check is entirely disabled for many languages that don't make a distinction between upper and lower case. Contact us if this is not yet disabled for your language.

### startpunc(\*args, \*\*kwargs)

Checks whether punctuation at the beginning of the strings match.

Operates as endpunc but you will probably see fewer errors.

## startwhitespace(\*args, \*\*kwargs)

Checks whether whitespace at the beginning of the strings matches.

As in endwhitespace but you will see fewer errors.

### tabs (\*args, \*\*kwargs)

Checks whether tabs are consistent between the two strings.

Counts the number of \t tab markers and reports an error if they differ.

## unchanged(\*args, \*\*kwargs)

Checks whether a translation is basically identical to the original string.

This checks to see if the translation isn't just a copy of the English original. Sometimes, this is what you want, but other times you will detect words that should have been translated.

### untranslated(\*args, \*\*kwargs)

Checks whether a string has been translated at all.

This check is really only useful if you want to extract untranslated strings so that they can be translated independently of the main work.

#### urls (\*args, \*\*kwargs)

Checks that URLs are not translated.

This checks only basic URLs (http, ftp, mailto etc.) not all URIs (e.g. afp, smb, file). Generally, you don't want to translate URLs, unless they are example URLs (http://your\_server.com/filename.html). If the URL is for configuration information, then you need to query the developers about placing configuration information in PO files. It shouldn't really be there, unless it is very clearly marked: such information should go into a configuration file.

## validchars (\*args, \*\*kwargs)

Checks that only characters specified as valid appear in the translation.

Often during character conversion to and from UTF-8 you get some strange characters appearing in your translation. This test presents a simple way to try and identify such errors.

This test will only run of you specify the --validcharsfile command line option. This file contains all the characters that are valid in your language. You must use UTF-8 encoding for the characters in the file.

If the test finds any characters not in your valid characters file then the test will print the character together with its Unicode value (e.g. 002B).

### validxml(\*args, \*\*kwargs)

Check that all XML/HTML open/close tags has close/open pair in the translation.

# variables (\*args, \*\*kwargs)

Checks whether variables of various forms are consistent between the two strings.

This checks to make sure that variables that appear in the original also appear in the translation. It can handle variables from projects like KDE or OpenOffice. It does not at the moment cope with variables that use the reordering syntax of Gettext PO files.

## xmltags (\*args, \*\*kwargs)

Checks that XML/HTML tags have not been translated.

This check finds the number of tags in the source string and checks that the same number are in the translation. If the counts don't match then either the tag is missing or it was mistakenly translated by the translator, both of which are errors.

The check ignores tags or things that look like tags that cover the whole string e.g. <Error> but will produce false positives for things like An <Error> occurred as here Error should be translated. It also will allow translation of the *alt* attribute in e.g. <img src="bob.png" alt="Image description"> or similar translatable attributes in OpenOffice.org help files.

class translate.filters.checks.MinimalChecker(\*\*kwargs)

### accelerators (\*args, \*\*kwargs)

Checks whether accelerators are consistent between the two strings.

This test is capable of checking the different type of accelerators that are used in different projects, like Mozilla or KDE. The test will pick up accelerators that are missing and ones that shouldn't be there.

See accelerators on the localization guide for a full description on accelerators.

#### acronyms (\*args, \*\*kwargs)

Checks that acronyms that appear are unchanged.

If an acronym appears in the original this test will check that it appears in the translation. Translating acronyms is a language decision but many languages leave them unchanged. In that case this test is useful for tracking down translations of the acronym and correcting them.

#### blank (\*args, \*\*kwargs)

Checks whether a translation is totally blank.

This will check to see if a translation has inadvertently been translated as blank i.e. as spaces. This is different from untranslated which is completely empty. This test is useful in that if something is translated as "" it will appear to most tools as if it is translated.

## brackets (\*args, \*\*kwargs)

Checks that the number of brackets in both strings match.

If ( [ $\{ or \} ]$ ) appear in the original this will check that the same number appear in the translation.

#### checker\_name

Extract checker name, for example 'mozilla' from MozillaChecker.

#### compendiumconflicts(\*args, \*\*kwargs)

Checks for Gettext compendium conflicts (#-#-#-#).

When you use msgcat to create a PO compendium it will insert #-#-#-# into entries that are not consistent. If the compendium is used later in a message merge then these conflicts will appear in your translations. This test quickly extracts those for correction.

#### credits (\*args, \*\*kwargs)

Checks for messages containing translation credits instead of normal translations.

Some projects have consistent ways of giving credit to translators by having a unit or two where translators can fill in their name and possibly their contact details. This test allows you to find these units easily to check that they are completed correctly and also disables other tests that might incorrectly get triggered for these units (such as urls, emails, etc.)

### doublequoting(\*args, \*\*kwargs)

Checks whether doublequoting is consistent between the two strings.

Checks on double quotes " to ensure that you have the same number in both the original and the translated string. This tests takes into account that several languages use different quoting characters, and will test for them instead.

## doublespacing(\*args, \*\*kwargs)

Checks for bad double-spaces by comparing to original.

This will identify if you have [space][space] in when you don't have it in the original or it appears in the original but not in your translation. Some of these are spurious and how you correct them depends on the conventions of your language.

# doublewords (\*args, \*\*kwargs)

Checks for repeated words in the translation.

Words that have been repeated in a translation will be highlighted with this test e.g. "the the", "a a". These are generally typos that need correcting. Some languages may have valid repeated words in their structure, in that case either ignore those instances or switch this test off.

```
emails (*args, **kwargs)
```

Checks that emails are not translated.

Generally you should not be translating email addresses. This check will look to see that email addresses e.g. info@example.com are not translated. In some cases of course you should translate the address but generally you shouldn't.

### endpunc (\*args, \*\*kwargs)

Checks whether punctuation at the end of the strings match.

This will ensure that the ending of your translation has the same punctuation as the original. E.g. if it ends in :[space] then so should yours. It is useful for ensuring that you have ellipses [...] in all your translations, not simply three separate full-stops. You may pick up some errors in the original: feel free to keep your translation and notify the programmers. In some languages, characters such as ? or ! are always preceded by a space e.g. [space]? — do what your language customs dictate. Other false positives you will notice are, for example, if through changes in word-order you add "), etc. at the end of the sentence. Do not change these: your language word-order takes precedence.

It must be noted that if you are tempted to leave out [full-stop] or [colon] or add [full-stop] to a sentence, that often these have been done for a reason, e.g. a list where fullstops make it look cluttered. So, initially match them with the English, and make changes once the program is being used.

This check is aware of several language conventions for punctuation characters, such as the custom question marks for Greek and Arabic, Devanagari Danda, full-width punctuation for CJK languages, etc. Support for your language can be added easily if it is not there yet.

# endwhitespace(\*args, \*\*kwargs)

Checks whether whitespace at the end of the strings matches.

Operates the same as endpunc but is only concerned with whitespace. This filter is particularly useful for those strings which will evidently be followed by another string in the program, e.g. [Password: ] or [Enter your username: ]. The whitespace is an inherent part of the string. This filter makes sure you don't miss those important but otherwise invisible spaces!

If your language uses full-width punctuation (like Chinese), the visual spacing in the character might be enough without an added extra space.

### escapes (\*args, \*\*kwargs)

Checks whether escaping is consistent between the two strings.

Checks escapes such as  $\n \u0000$  to ensure that if they exist in the original string you also have them in the translation.

## filepaths (\*args, \*\*kwargs)

Checks that file paths have not been translated.

Checks that paths such as /home/user1 have not been translated. Generally you do not translate a file path, unless it is being used as an example, e.g. your\_user\_name/path/to/filename.conf.

#### filteraccelerators by list(strl, acceptlist=None)

Filter out accelerators from str1.

## functions (\*args, \*\*kwargs)

Checks that function names are not translated.

Checks that function names e.g. rgb() or getEntity.Name() are not translated.

#### get\_ignored\_filters()

Return checker's additional filters for current language.

## getfilters (excludefilters=None, limitfilters=None)

Returns dictionary of available filters, including/excluding those in the given lists.

#### kdecomments (\*args, \*\*kwargs)

Checks to ensure that no KDE style comments appear in the translation.

KDE style translator comments appear in PO files as "\_: comment\n". New translators often translate the comment. This test tries to identify instances where the comment has been translated.

#### long (\*args, \*\*kwargs)

Checks whether a translation is much longer than the original string.

This is most useful in the special case where the translation is multiple characters long while the source text is only 1 character long. Otherwise, we use a general ratio that will catch very big differences but is set conservatively to limit the number of false positives.

# musttranslatewords(\*args, \*\*kwargs)

Checks that words configured as definitely translatable don't appear in the translation.

If for instance in your language you decide that you must translate 'OK' then this test will flag any occurrences of 'OK' in the translation if it appeared in the source string. You must specify a file containing all of the *must translate* words using --musttranslatefile.

## newlines (\*args, \*\*kwargs)

Checks whether newlines are consistent between the two strings.

Counts the number of \n newlines (and variants such as \r\n) and reports and error if they differ.

## notranslatewords (\*args, \*\*kwargs)

Checks that words configured as untranslatable appear in the translation too.

Many brand names should not be translated, this test allows you to easily make sure that words like: Word, Excel, Impress, Calc, etc. are not translated. You must specify a file containing all of the *no translate* words using --notranslatefile.

#### numbers (\*args, \*\*kwargs)

Checks whether numbers of various forms are consistent between the two strings.

You will see some errors where you have either written the number in full or converted it to the digit in your translation. Also changes in order will trigger this error.

## options (\*args, \*\*kwargs)

Checks that command line options are not translated.

In messages that contain command line options, such as <code>--help</code>, this test will check that these remain untranslated. These could be translated in the future if programs can create a mechanism to allow this, but currently they are not translated. If the options has a parameter, e.g. <code>--file=FILE</code>, then the test will check that the parameter has been translated.

# printf(\*args, \*\*kwargs)

Checks whether printf format strings match.

If the printf formatting variables are not identical, then this will indicate an error. Printf statements are used by programs to format output in a human readable form (they are placeholders for variable data). They allow you to specify lengths of string variables, string padding, number padding, precision, etc. Generally they will look like this: %d, %5.2f, %100s, etc. The test can also manage variables-reordering using the %1\$s syntax. The variables' type and details following data are tested to ensure that they are strictly identical, but they may be reordered.

See also printf Format String.

#### puncspacing(\*args, \*\*kwargs)

Checks for bad spacing after punctuation.

In the case of [full-stop][space] in the original, this test checks that your translation does not remove the space. It checks also for [comma], [colon], etc.

Some languages don't use spaces after common punctuation marks, especially where full-width punctuation marks are used. This check will take that into account.

#### purepunc (\*args, \*\*kwargs)

Checks that strings that are purely punctuation are not changed.

This extracts strings like + or - as these usually should not be changed.

### pythonbraceformat (\*args, \*\*kwargs)

Checks whether python brace format strings match.

#### run\_filters (unit, categorised=False)

Do some optimisation by caching some data of the unit for the benefit of run test ().

#### run\_test (test, unit)

Runs the given test on the given unit.

Note that this can raise a *FilterFailure* as part of normal operation.

#### sentencecount (\*args, \*\*kwargs)

Checks that the number of sentences in both strings match.

Adds the number of sentences to see that the sentence count is the same between the original and translated string. You may not always want to use this test, if you find you often need to reformat your translation, because the original is badly-expressed, or because the structure of your language works better that way. Do what works best for your language: it's the meaning of the original you want to convey, not the exact way it was written in the English.

#### setconfig(config)

Sets the accelerator list.

#### setsuggestionstore(store)

Sets the filename that a checker should use for evaluating suggestions.

```
short (*args, **kwargs)
```

Checks whether a translation is much shorter than the original string.

This is most useful in the special case where the translation is 1 characters long while the source text is multiple characters long. Otherwise, we use a general ratio that will catch very big differences but is set conservatively to limit the number of false positives.

```
simplecaps (*args, **kwargs)
```

Checks the capitalisation of two strings isn't wildly different.

This will pick up many false positives, so don't be a slave to it. It is useful for identifying translations that don't start with a capital letter (upper-case letter) when they should, or those that do when they shouldn't. It will also highlight sentences that have extra capitals; depending on the capitalisation convention of your language, you might want to change these to Title Case, or change them all to normal sentence case.

### simpleplurals(\*args, \*\*kwargs)

Checks for English style plural(s) for you to review.

This test will extract any message that contains words with a final "(s)" in the source text. You can then inspect the message, to check that the correct plural form has been used for your language. In some languages, plurals are made by adding text at the beginning of words, making the English style messy. In this case, they often revert to the plural form. This test allows an editor to check that the plurals used are correct. Be aware that this test may create a number of false positives.

For languages with no plural forms (only one noun form) this test will simply test that nothing like "(s)" was used in the translation.

## singlequoting(\*args, \*\*kwargs)

Checks whether singlequoting is consistent between the two strings.

The same as doublequoting but checks for the 'character. Because this is used in contractions like it's and in possessive forms like user's, this test can output spurious errors if your language doesn't use such

forms. If a quote appears at the end of a sentence in the translation, i.e. '., this might not be detected properly by the check.

### spellcheck (\*args, \*\*kwargs)

Checks words that don't pass a spell check.

This test will check for misspelled words in your translation. The test first checks for misspelled words in the original (usually English) text, and adds those to an exclusion list. The advantage of this exclusion is that many words that are specific to the application will not raise errors e.g. program names, brand names, function names.

The checker works with PyEnchant. You need to have PyEnchant installed as well as a dictionary for your language (for example, one of the Hunspell or aspell dictionaries). This test will only work if you have specified the --language option.

The pofilter error that is created, lists the misspelled word, plus suggestions returned from the spell checker. That makes it easy for you to identify the word and select a replacement.

# startcaps (\*args, \*\*kwargs)

Checks that the message starts with the correct capitalisation.

After stripping whitespace and common punctuation characters, it then checks to see that the first remaining character is correctly capitalised. So, if the sentence starts with an upper-case letter, and the translation does not, an error is produced.

This check is entirely disabled for many languages that don't make a distinction between upper and lower case. Contact us if this is not yet disabled for your language.

#### startpunc(\*args, \*\*kwargs)

Checks whether punctuation at the beginning of the strings match.

Operates as endpunc but you will probably see fewer errors.

# startwhitespace(\*args, \*\*kwargs)

Checks whether whitespace at the beginning of the strings matches.

As in endwhitespace but you will see fewer errors.

### tabs (\*args, \*\*kwargs)

Checks whether tabs are consistent between the two strings.

Counts the number of \t tab markers and reports an error if they differ.

### unchanged (\*args, \*\*kwargs)

Checks whether a translation is basically identical to the original string.

This checks to see if the translation isn't just a copy of the English original. Sometimes, this is what you want, but other times you will detect words that should have been translated.

#### untranslated(\*args, \*\*kwargs)

Checks whether a string has been translated at all.

This check is really only useful if you want to extract untranslated strings so that they can be translated independently of the main work.

### urls (\*args, \*\*kwargs)

Checks that URLs are not translated.

This checks only basic URLs (http, ftp, mailto etc.) not all URIs (e.g. afp, smb, file). Generally, you don't want to translate URLs, unless they are example URLs (http://your\_server.com/filename.html). If the URL is for configuration information, then you need to query the developers about placing configuration information in PO files. It shouldn't really be there, unless it is very clearly marked: such information should go into a configuration file.

#### validchars (\*args, \*\*kwargs)

Checks that only characters specified as valid appear in the translation.

Often during character conversion to and from UTF-8 you get some strange characters appearing in your translation. This test presents a simple way to try and identify such errors.

This test will only run of you specify the --validcharsfile command line option. This file contains all the characters that are valid in your language. You must use UTF-8 encoding for the characters in the file.

If the test finds any characters not in your valid characters file then the test will print the character together with its Unicode value (e.g. 002B).

## variables (\*args, \*\*kwargs)

Checks whether variables of various forms are consistent between the two strings.

This checks to make sure that variables that appear in the original also appear in the translation. It can handle variables from projects like KDE or OpenOffice. It does not at the moment cope with variables that use the reordering syntax of Gettext PO files.

#### xmltags (\*args, \*\*kwargs)

Checks that XML/HTML tags have not been translated.

This check finds the number of tags in the source string and checks that the same number are in the translation. If the counts don't match then either the tag is missing or it was mistakenly translated by the translator, both of which are errors.

The check ignores tags or things that look like tags that cover the whole string e.g. <Error> but will produce false positives for things like An <Error> occurred as here Error should be translated. It also will allow translation of the *alt* attribute in e.g. <img src="bob.png" alt="Image description"> or similar translatable attributes in OpenOffice.org help files.

class translate.filters.checks.MozillaChecker(\*\*kwargs)

### accelerators (\*args, \*\*kwargs)

Checks whether accelerators are consistent between the two strings.

For Mozilla we lower the severity to cosmetic, and for some languages it also ensures accelerators are absent in the target string since some languages do not use accelerators, for example Indic languages.

## acronyms (\*args, \*\*kwargs)

Checks that acronyms that appear are unchanged.

If an acronym appears in the original this test will check that it appears in the translation. Translating acronyms is a language decision but many languages leave them unchanged. In that case this test is useful for tracking down translations of the acronym and correcting them.

#### blank (\*args, \*\*kwargs)

Checks whether a translation is totally blank.

This will check to see if a translation has inadvertently been translated as blank i.e. as spaces. This is different from untranslated which is completely empty. This test is useful in that if something is translated as "" it will appear to most tools as if it is translated.

#### brackets (\*args, \*\*kwargs)

Checks that the number of brackets in both strings match.

If ([{ or }]) appear in the original this will check that the same number appear in the translation.

### checker name

Extract checker name, for example 'mozilla' from MozillaChecker.

#### compendiumconflicts(\*args, \*\*kwargs)

Checks for Gettext compendium conflicts (#-#-#-#).

When you use msgcat to create a PO compendium it will insert #-#-#-# into entries that are not consistent. If the compendium is used later in a message merge then these conflicts will appear in your translations. This test quickly extracts those for correction.

#### credits (\*args, \*\*kwargs)

Checks for messages containing translation credits instead of normal translations.

Some projects have consistent ways of giving credit to translators by having a unit or two where translators can fill in their name and possibly their contact details. This test allows you to find these units easily to check that they are completed correctly and also disables other tests that might incorrectly get triggered for these units (such as urls, emails, etc.)

#### dialogsizes(\*args, \*\*kwargs)

Checks that dialog sizes are not translated.

This is a Mozilla specific test. Mozilla uses a language called XUL to define dialogues and screens. This can make use of CSS to specify properties of the dialogue. These properties include things such as the width and height of the box. The size might need to be changed if the dialogue size changes due to longer translations. Thus translators can change these settings. But you are only meant to change the number not translate the words 'width' or 'height'. This check capture instances where these are translated. It will also catch other types of errors in these units.

## doublequoting(\*args, \*\*kwargs)

Checks whether doublequoting is consistent between the two strings.

Checks on double quotes " to ensure that you have the same number in both the original and the translated string. This tests takes into account that several languages use different quoting characters, and will test for them instead.

## doublespacing(\*args, \*\*kwargs)

Checks for bad double-spaces by comparing to original.

This will identify if you have [space][space] in when you don't have it in the original or it appears in the original but not in your translation. Some of these are spurious and how you correct them depends on the conventions of your language.

## doublewords (\*args, \*\*kwargs)

Checks for repeated words in the translation.

Words that have been repeated in a translation will be highlighted with this test e.g. "the the", "a a". These are generally typos that need correcting. Some languages may have valid repeated words in their structure, in that case either ignore those instances or switch this test off.

#### emails (\*args, \*\*kwargs)

Checks that emails are not translated.

Generally you should not be translating email addresses. This check will look to see that email addresses e.g. info@example.com are not translated. In some cases of course you should translate the address but generally you shouldn't.

### endpunc (\*args, \*\*kwargs)

Checks whether punctuation at the end of the strings match.

This will ensure that the ending of your translation has the same punctuation as the original. E.g. if it ends in :[space] then so should yours. It is useful for ensuring that you have ellipses [...] in all your translations, not simply three separate full-stops. You may pick up some errors in the original: feel free to keep your translation and notify the programmers. In some languages, characters such as ? or ! are always preceded by a space e.g. [space]? — do what your language customs dictate. Other false positives you

will notice are, for example, if through changes in word-order you add "), etc. at the end of the sentence. Do not change these: your language word-order takes precedence.

It must be noted that if you are tempted to leave out [full-stop] or [colon] or add [full-stop] to a sentence, that often these have been done for a reason, e.g. a list where fullstops make it look cluttered. So, initially match them with the English, and make changes once the program is being used.

This check is aware of several language conventions for punctuation characters, such as the custom question marks for Greek and Arabic, Devanagari Danda, full-width punctuation for CJK languages, etc. Support for your language can be added easily if it is not there yet.

### endwhitespace(\*args, \*\*kwargs)

Checks whether whitespace at the end of the strings matches.

Operates the same as endpunc but is only concerned with whitespace. This filter is particularly useful for those strings which will evidently be followed by another string in the program, e.g. [Password: ] or [Enter your username: ]. The whitespace is an inherent part of the string. This filter makes sure you don't miss those important but otherwise invisible spaces!

If your language uses full-width punctuation (like Chinese), the visual spacing in the character might be enough without an added extra space.

#### escapes (\*args, \*\*kwargs)

Checks whether escaping is consistent between the two strings.

Checks escapes such as  $\n \u0000$  to ensure that if they exist in the original string you also have them in the translation.

# filepaths (\*args, \*\*kwargs)

Checks that file paths have not been translated.

Checks that paths such as /home/user1 have not been translated. Generally you do not translate a file path, unless it is being used as an example, e.g. your\_user\_name/path/to/filename.conf.

## filteraccelerators\_by\_list(str1, acceptlist=None)

Filter out accelerators from str1.

### functions (\*args, \*\*kwargs)

Checks that function names are not translated.

Checks that function names e.g. rgb() or getEntity.Name() are not translated.

## get ignored filters()

Return checker's additional filters for current language.

### getfilters (excludefilters=None, limitfilters=None)

Returns dictionary of available filters, including/excluding those in the given lists.

### kdecomments (\*args, \*\*kwargs)

Checks to ensure that no KDE style comments appear in the translation.

KDE style translator comments appear in PO files as "\_: comment\n". New translators often translate the comment. This test tries to identify instances where the comment has been translated.

#### long (\*args, \*\*kwargs)

Checks whether a translation is much longer than the original string.

This is most useful in the special case where the translation is multiple characters long while the source text is only 1 character long. Otherwise, we use a general ratio that will catch very big differences but is set conservatively to limit the number of false positives.

#### musttranslatewords(\*args, \*\*kwargs)

Checks that words configured as definitely translatable don't appear in the translation.

If for instance in your language you decide that you must translate 'OK' then this test will flag any occurrences of 'OK' in the translation if it appeared in the source string. You must specify a file containing all of the *must translate* words using --musttranslatefile.

### newlines (\*args, \*\*kwargs)

Checks whether newlines are consistent between the two strings.

Counts the number of \n newlines (and variants such as \r\n) and reports and error if they differ.

#### notranslatewords (\*args, \*\*kwargs)

Checks that words configured as untranslatable appear in the translation too.

Many brand names should not be translated, this test allows you to easily make sure that words like: Word, Excel, Impress, Calc, etc. are not translated. You must specify a file containing all of the *no translate* words using --notranslatefile.

```
numbers (*args, **kwargs)
```

Checks that numbers are not translated.

Special handling for Mozilla to ignore entries that are dialog sizes.

```
options (*args, **kwargs)
```

Checks that command line options are not translated.

In messages that contain command line options, such as <code>--help</code>, this test will check that these remain untranslated. These could be translated in the future if programs can create a mechanism to allow this, but currently they are not translated. If the options has a parameter, e.g. <code>--file=FILE</code>, then the test will check that the parameter has been translated.

```
printf(*args, **kwargs)
```

Checks whether printf format strings match.

If the printf formatting variables are not identical, then this will indicate an error. Printf statements are used by programs to format output in a human readable form (they are placeholders for variable data). They allow you to specify lengths of string variables, string padding, number padding, precision, etc. Generally they will look like this: %d, %5.2f, %100s, etc. The test can also manage variables-reordering using the %1\$s syntax. The variables' type and details following data are tested to ensure that they are strictly identical, but they may be reordered.

See also printf Format String.

## puncspacing(\*args, \*\*kwargs)

Checks for bad spacing after punctuation.

In the case of [full-stop][space] in the original, this test checks that your translation does not remove the space. It checks also for [comma], [colon], etc.

Some languages don't use spaces after common punctuation marks, especially where full-width punctuation marks are used. This check will take that into account.

```
purepunc (*args, **kwargs)
```

Checks that strings that are purely punctuation are not changed.

This extracts strings like + or - as these usually should not be changed.

### pythonbraceformat (\*args, \*\*kwargs)

Checks whether python brace format strings match.

#### run\_filters (unit, categorised=False)

Do some optimisation by caching some data of the unit for the benefit of run\_test().

#### run test(test, unit)

Runs the given test on the given unit.

Note that this can raise a *FilterFailure* as part of normal operation.

#### sentencecount (\*args, \*\*kwargs)

Checks that the number of sentences in both strings match.

Adds the number of sentences to see that the sentence count is the same between the original and translated string. You may not always want to use this test, if you find you often need to reformat your translation, because the original is badly-expressed, or because the structure of your language works better that way. Do what works best for your language: it's the meaning of the original you want to convey, not the exact way it was written in the English.

#### setconfig(config)

Sets the accelerator list.

#### setsuggestionstore(store)

Sets the filename that a checker should use for evaluating suggestions.

### short (\*args, \*\*kwargs)

Checks whether a translation is much shorter than the original string.

This is most useful in the special case where the translation is 1 characters long while the source text is multiple characters long. Otherwise, we use a general ratio that will catch very big differences but is set conservatively to limit the number of false positives.

## simplecaps (\*args, \*\*kwargs)

Checks the capitalisation of two strings isn't wildly different.

This will pick up many false positives, so don't be a slave to it. It is useful for identifying translations that don't start with a capital letter (upper-case letter) when they should, or those that do when they shouldn't. It will also highlight sentences that have extra capitals; depending on the capitalisation convention of your language, you might want to change these to Title Case, or change them all to normal sentence case.

## simpleplurals(\*args, \*\*kwargs)

Checks for English style plural(s) for you to review.

This test will extract any message that contains words with a final "(s)" in the source text. You can then inspect the message, to check that the correct plural form has been used for your language. In some languages, plurals are made by adding text at the beginning of words, making the English style messy. In this case, they often revert to the plural form. This test allows an editor to check that the plurals used are correct. Be aware that this test may create a number of false positives.

For languages with no plural forms (only one noun form) this test will simply test that nothing like "(s)" was used in the translation.

### singlequoting(\*args, \*\*kwargs)

Checks whether singlequoting is consistent between the two strings.

The same as doublequoting but checks for the 'character. Because this is used in contractions like it's and in possessive forms like user's, this test can output spurious errors if your language doesn't use such forms. If a quote appears at the end of a sentence in the translation, i.e. '., this might not be detected properly by the check.

# spellcheck (\*args, \*\*kwargs)

Checks words that don't pass a spell check.

This test will check for misspelled words in your translation. The test first checks for misspelled words in the original (usually English) text, and adds those to an exclusion list. The advantage of this exclusion is that many words that are specific to the application will not raise errors e.g. program names, brand names, function names.

The checker works with PyEnchant. You need to have PyEnchant installed as well as a dictionary for your language (for example, one of the Hunspell or aspell dictionaries). This test will only work if you have

specified the --language option.

The pofilter error that is created, lists the misspelled word, plus suggestions returned from the spell checker. That makes it easy for you to identify the word and select a replacement.

## startcaps (\*args, \*\*kwargs)

Checks that the message starts with the correct capitalisation.

After stripping whitespace and common punctuation characters, it then checks to see that the first remaining character is correctly capitalised. So, if the sentence starts with an upper-case letter, and the translation does not, an error is produced.

This check is entirely disabled for many languages that don't make a distinction between upper and lower case. Contact us if this is not yet disabled for your language.

## startpunc (\*args, \*\*kwargs)

Checks whether punctuation at the beginning of the strings match.

Operates as endpunc but you will probably see fewer errors.

# startwhitespace(\*args, \*\*kwargs)

Checks whether whitespace at the beginning of the strings matches.

As in endwhitespace but you will see fewer errors.

# tabs (\*args, \*\*kwargs)

Checks whether tabs are consistent between the two strings.

Counts the number of \t tab markers and reports an error if they differ.

#### unchanged(\*args, \*\*kwargs)

Checks whether a translation is basically identical to the original string.

Special handling for Mozilla to ignore entries that are dialog sizes.

## untranslated(\*args, \*\*kwargs)

Checks whether a string has been translated at all.

This check is really only useful if you want to extract untranslated strings so that they can be translated independently of the main work.

# urls (\*args, \*\*kwargs)

Checks that URLs are not translated.

This checks only basic URLs (http, ftp, mailto etc.) not all URIs (e.g. afp, smb, file). Generally, you don't want to translate URLs, unless they are example URLs (http://your\_server.com/filename.html). If the URL is for configuration information, then you need to query the developers about placing configuration information in PO files. It shouldn't really be there, unless it is very clearly marked: such information should go into a configuration file.

#### validchars (\*args, \*\*kwargs)

Checks that only characters specified as valid appear in the translation.

Often during character conversion to and from UTF-8 you get some strange characters appearing in your translation. This test presents a simple way to try and identify such errors.

This test will only run of you specify the --validcharsfile command line option. This file contains all the characters that are valid in your language. You must use UTF-8 encoding for the characters in the file.

If the test finds any characters not in your valid characters file then the test will print the character together with its Unicode value (e.g. 002B).

#### variables (\*args, \*\*kwargs)

Checks whether variables of various forms are consistent between the two strings.

This checks to make sure that variables that appear in the original also appear in the translation. It can handle variables from projects like KDE or OpenOffice. It does not at the moment cope with variables that use the reordering syntax of Gettext PO files.

```
xmltags (*args, **kwargs)
```

Checks that XML/HTML tags have not been translated.

This check finds the number of tags in the source string and checks that the same number are in the translation. If the counts don't match then either the tag is missing or it was mistakenly translated by the translator, both of which are errors.

The check ignores tags or things that look like tags that cover the whole string e.g. <Error> but will produce false positives for things like An <Error> occurred as here Error should be translated. It also will allow translation of the *alt* attribute in e.g. <img src="bob.png" alt="Image description"> or similar translatable attributes in OpenOffice.org help files.

class translate.filters.checks.OpenOfficeChecker(\*\*kwargs)

#### accelerators (\*args, \*\*kwargs)

Checks whether accelerators are consistent between the two strings.

This test is capable of checking the different type of accelerators that are used in different projects, like Mozilla or KDE. The test will pick up accelerators that are missing and ones that shouldn't be there.

See accelerators on the localization guide for a full description on accelerators.

```
acronyms (*args, **kwargs)
```

Checks that acronyms that appear are unchanged.

If an acronym appears in the original this test will check that it appears in the translation. Translating acronyms is a language decision but many languages leave them unchanged. In that case this test is useful for tracking down translations of the acronym and correcting them.

### blank (\*args, \*\*kwargs)

Checks whether a translation is totally blank.

This will check to see if a translation has inadvertently been translated as blank i.e. as spaces. This is different from untranslated which is completely empty. This test is useful in that if something is translated as "" it will appear to most tools as if it is translated.

```
brackets (*args, **kwargs)
```

Checks that the number of brackets in both strings match.

If ([{ or }]) appear in the original this will check that the same number appear in the translation.

### checker\_name

Extract checker name, for example 'mozilla' from MozillaChecker.

# compendiumconflicts(\*args, \*\*kwargs)

Checks for Gettext compendium conflicts (#-#-#-#).

When you use msgcat to create a PO compendium it will insert #-#-#-#-# into entries that are not consistent. If the compendium is used later in a message merge then these conflicts will appear in your translations. This test quickly extracts those for correction.

## credits(\*args, \*\*kwargs)

Checks for messages containing translation credits instead of normal translations.

Some projects have consistent ways of giving credit to translators by having a unit or two where translators can fill in their name and possibly their contact details. This test allows you to find these units easily to check that they are completed correctly and also disables other tests that might incorrectly get triggered for these units (such as urls, emails, etc.)

### doublequoting(\*args, \*\*kwargs)

Checks whether doublequoting is consistent between the two strings.

Checks on double quotes " to ensure that you have the same number in both the original and the translated string. This tests takes into account that several languages use different quoting characters, and will test for them instead.

### doublespacing(\*args, \*\*kwargs)

Checks for bad double-spaces by comparing to original.

This will identify if you have [space][space] in when you don't have it in the original or it appears in the original but not in your translation. Some of these are spurious and how you correct them depends on the conventions of your language.

### doublewords (\*args, \*\*kwargs)

Checks for repeated words in the translation.

Words that have been repeated in a translation will be highlighted with this test e.g. "the the", "a a". These are generally typos that need correcting. Some languages may have valid repeated words in their structure, in that case either ignore those instances or switch this test off.

# emails (\*args, \*\*kwargs)

Checks that emails are not translated.

Generally you should not be translating email addresses. This check will look to see that email addresses e.g. info@example.com are not translated. In some cases of course you should translate the address but generally you shouldn't.

### endpunc (\*args, \*\*kwargs)

Checks whether punctuation at the end of the strings match.

This will ensure that the ending of your translation has the same punctuation as the original. E.g. if it ends in :[space] then so should yours. It is useful for ensuring that you have ellipses [...] in all your translations, not simply three separate full-stops. You may pick up some errors in the original: feel free to keep your translation and notify the programmers. In some languages, characters such as ? or ! are always preceded by a space e.g. [space]? — do what your language customs dictate. Other false positives you will notice are, for example, if through changes in word-order you add "), etc. at the end of the sentence. Do not change these: your language word-order takes precedence.

It must be noted that if you are tempted to leave out [full-stop] or [colon] or add [full-stop] to a sentence, that often these have been done for a reason, e.g. a list where fullstops make it look cluttered. So, initially match them with the English, and make changes once the program is being used.

This check is aware of several language conventions for punctuation characters, such as the custom question marks for Greek and Arabic, Devanagari Danda, full-width punctuation for CJK languages, etc. Support for your language can be added easily if it is not there yet.

### endwhitespace(\*args, \*\*kwargs)

Checks whether whitespace at the end of the strings matches.

Operates the same as endpunc but is only concerned with whitespace. This filter is particularly useful for those strings which will evidently be followed by another string in the program, e.g. [Password: ] or [Enter your username: ]. The whitespace is an inherent part of the string. This filter makes sure you don't miss those important but otherwise invisible spaces!

If your language uses full-width punctuation (like Chinese), the visual spacing in the character might be enough without an added extra space.

### escapes (\*args, \*\*kwargs)

Checks whether escaping is consistent between the two strings.

Checks escapes such as \n \u00000 to ensure that if they exist in the original string you also have them in the translation.

### filepaths (\*args, \*\*kwargs)

Checks that file paths have not been translated.

Checks that paths such as /home/user1 have not been translated. Generally you do not translate a file path, unless it is being used as an example, e.g. your\_user\_name/path/to/filename.conf.

### filteraccelerators\_by\_list(str1, acceptlist=None)

Filter out accelerators from str1.

## functions (\*args, \*\*kwargs)

Checks that function names are not translated.

Checks that function names e.g. rgb() or getEntity.Name() are not translated.

#### get\_ignored\_filters()

Return checker's additional filters for current language.

### getfilters (excludefilters=None, limitfilters=None)

Returns dictionary of available filters, including/excluding those in the given lists.

#### kdecomments (\*args, \*\*kwargs)

Checks to ensure that no KDE style comments appear in the translation.

KDE style translator comments appear in PO files as "\_: comment\n". New translators often translate the comment. This test tries to identify instances where the comment has been translated.

# long(\*args, \*\*kwargs)

Checks whether a translation is much longer than the original string.

This is most useful in the special case where the translation is multiple characters long while the source text is only 1 character long. Otherwise, we use a general ratio that will catch very big differences but is set conservatively to limit the number of false positives.

## musttranslatewords(\*args, \*\*kwargs)

Checks that words configured as definitely translatable don't appear in the translation.

If for instance in your language you decide that you must translate 'OK' then this test will flag any occurrences of 'OK' in the translation if it appeared in the source string. You must specify a file containing all of the *must translate* words using --musttranslatefile.

### newlines (\*args, \*\*kwargs)

Checks whether newlines are consistent between the two strings.

Counts the number of \n newlines (and variants such as \r\n) and reports and error if they differ.

### notranslatewords (\*args, \*\*kwargs)

Checks that words configured as untranslatable appear in the translation too.

Many brand names should not be translated, this test allows you to easily make sure that words like: Word, Excel, Impress, Calc, etc. are not translated. You must specify a file containing all of the *no translate* words using --notranslatefile.

## numbers (\*args, \*\*kwargs)

Checks whether numbers of various forms are consistent between the two strings.

You will see some errors where you have either written the number in full or converted it to the digit in your translation. Also changes in order will trigger this error.

```
options (*args, **kwargs)
```

Checks that command line options are not translated.

In messages that contain command line options, such as <code>--help</code>, this test will check that these remain untranslated. These could be translated in the future if programs can create a mechanism to allow this, but currently they are not translated. If the options has a parameter, e.g. <code>--file=FILE</code>, then the test will check that the parameter has been translated.

```
printf(*args, **kwargs)
```

Checks whether printf format strings match.

If the printf formatting variables are not identical, then this will indicate an error. Printf statements are used by programs to format output in a human readable form (they are placeholders for variable data). They allow you to specify lengths of string variables, string padding, number padding, precision, etc. Generally they will look like this: %d, %5.2f, %100s, etc. The test can also manage variables-reordering using the %1\$s syntax. The variables' type and details following data are tested to ensure that they are strictly identical, but they may be reordered.

See also printf Format String.

## puncspacing(\*args, \*\*kwargs)

Checks for bad spacing after punctuation.

In the case of [full-stop][space] in the original, this test checks that your translation does not remove the space. It checks also for [comma], [colon], etc.

Some languages don't use spaces after common punctuation marks, especially where full-width punctuation marks are used. This check will take that into account.

## purepunc (\*args, \*\*kwargs)

Checks that strings that are purely punctuation are not changed.

This extracts strings like + or - as these usually should not be changed.

### pythonbraceformat (\*args, \*\*kwargs)

Checks whether python brace format strings match.

#### run\_filters (unit, categorised=False)

Do some optimisation by caching some data of the unit for the benefit of run\_test().

## run\_test (test, unit)

Runs the given test on the given unit.

Note that this can raise a *FilterFailure* as part of normal operation.

### sentencecount (\*args, \*\*kwargs)

Checks that the number of sentences in both strings match.

Adds the number of sentences to see that the sentence count is the same between the original and translated string. You may not always want to use this test, if you find you often need to reformat your translation, because the original is badly-expressed, or because the structure of your language works better that way. Do what works best for your language: it's the meaning of the original you want to convey, not the exact way it was written in the English.

#### setconfig(config)

Sets the accelerator list.

## setsuggestionstore(store)

Sets the filename that a checker should use for evaluating suggestions.

#### short (\*args, \*\*kwargs)

Checks whether a translation is much shorter than the original string.

This is most useful in the special case where the translation is 1 characters long while the source text is multiple characters long. Otherwise, we use a general ratio that will catch very big differences but is set conservatively to limit the number of false positives.

#### simplecaps (\*args, \*\*kwargs)

Checks the capitalisation of two strings isn't wildly different.

This will pick up many false positives, so don't be a slave to it. It is useful for identifying translations that don't start with a capital letter (upper-case letter) when they should, or those that do when they shouldn't. It will also highlight sentences that have extra capitals; depending on the capitalisation convention of your language, you might want to change these to Title Case, or change them all to normal sentence case.

### simpleplurals(\*args, \*\*kwargs)

Checks for English style plural(s) for you to review.

This test will extract any message that contains words with a final "(s)" in the source text. You can then inspect the message, to check that the correct plural form has been used for your language. In some languages, plurals are made by adding text at the beginning of words, making the English style messy. In this case, they often revert to the plural form. This test allows an editor to check that the plurals used are correct. Be aware that this test may create a number of false positives.

For languages with no plural forms (only one noun form) this test will simply test that nothing like "(s)" was used in the translation.

## singlequoting(\*args, \*\*kwargs)

Checks whether singlequoting is consistent between the two strings.

The same as doublequoting but checks for the 'character. Because this is used in contractions like it's and in possessive forms like user's, this test can output spurious errors if your language doesn't use such forms. If a quote appears at the end of a sentence in the translation, i.e. '., this might not be detected properly by the check.

### spellcheck (\*args, \*\*kwargs)

Checks words that don't pass a spell check.

This test will check for misspelled words in your translation. The test first checks for misspelled words in the original (usually English) text, and adds those to an exclusion list. The advantage of this exclusion is that many words that are specific to the application will not raise errors e.g. program names, brand names, function names.

The checker works with PyEnchant. You need to have PyEnchant installed as well as a dictionary for your language (for example, one of the Hunspell or aspell dictionaries). This test will only work if you have specified the --language option.

The pofilter error that is created, lists the misspelled word, plus suggestions returned from the spell checker. That makes it easy for you to identify the word and select a replacement.

# startcaps (\*args, \*\*kwargs)

Checks that the message starts with the correct capitalisation.

After stripping whitespace and common punctuation characters, it then checks to see that the first remaining character is correctly capitalised. So, if the sentence starts with an upper-case letter, and the translation does not, an error is produced.

This check is entirely disabled for many languages that don't make a distinction between upper and lower case. Contact us if this is not yet disabled for your language.

### startpunc(\*args, \*\*kwargs)

Checks whether punctuation at the beginning of the strings match.

Operates as endpunc but you will probably see fewer errors.

#### startwhitespace (\*args, \*\*kwargs)

Checks whether whitespace at the beginning of the strings matches.

As in endwhitespace but you will see fewer errors.

#### tabs (\*args, \*\*kwargs)

Checks whether tabs are consistent between the two strings.

Counts the number of \t tab markers and reports an error if they differ.

### unchanged(\*args, \*\*kwargs)

Checks whether a translation is basically identical to the original string.

This checks to see if the translation isn't just a copy of the English original. Sometimes, this is what you want, but other times you will detect words that should have been translated.

### untranslated(\*args, \*\*kwargs)

Checks whether a string has been translated at all.

This check is really only useful if you want to extract untranslated strings so that they can be translated independently of the main work.

### urls (\*args, \*\*kwargs)

Checks that URLs are not translated.

This checks only basic URLs (http, ftp, mailto etc.) not all URIs (e.g. afp, smb, file). Generally, you don't want to translate URLs, unless they are example URLs (http://your\_server.com/filename.html). If the URL is for configuration information, then you need to query the developers about placing configuration information in PO files. It shouldn't really be there, unless it is very clearly marked: such information should go into a configuration file.

## validchars (\*args, \*\*kwargs)

Checks that only characters specified as valid appear in the translation.

Often during character conversion to and from UTF-8 you get some strange characters appearing in your translation. This test presents a simple way to try and identify such errors.

This test will only run of you specify the --validcharsfile command line option. This file contains all the characters that are valid in your language. You must use UTF-8 encoding for the characters in the file.

If the test finds any characters not in your valid characters file then the test will print the character together with its Unicode value (e.g. 002B).

### variables (\*args, \*\*kwargs)

Checks whether variables of various forms are consistent between the two strings.

This checks to make sure that variables that appear in the original also appear in the translation. It can handle variables from projects like KDE or OpenOffice. It does not at the moment cope with variables that use the reordering syntax of Gettext PO files.

# xmltags (\*args, \*\*kwargs)

Checks that XML/HTML tags have not been translated.

This check finds the number of tags in the source string and checks that the same number are in the translation. If the counts don't match then either the tag is missing or it was mistakenly translated by the translator, both of which are errors.

The check ignores tags or things that look like tags that cover the whole string e.g. <Error> but will produce false positives for things like An <Error> occurred as here Error should be trans-

lated. It also will allow translation of the *alt* attribute in e.g. <img src="bob.png" alt="Image description"> or similar translatable attributes in OpenOffice.org help files.

class translate.filters.checks.ReducedChecker(\*\*kwargs)

### accelerators (\*args, \*\*kwargs)

Checks whether accelerators are consistent between the two strings.

This test is capable of checking the different type of accelerators that are used in different projects, like Mozilla or KDE. The test will pick up accelerators that are missing and ones that shouldn't be there.

See accelerators on the localization guide for a full description on accelerators.

```
acronyms (*args, **kwargs)
```

Checks that acronyms that appear are unchanged.

If an acronym appears in the original this test will check that it appears in the translation. Translating acronyms is a language decision but many languages leave them unchanged. In that case this test is useful for tracking down translations of the acronym and correcting them.

```
blank (*args, **kwargs)
```

Checks whether a translation is totally blank.

This will check to see if a translation has inadvertently been translated as blank i.e. as spaces. This is different from untranslated which is completely empty. This test is useful in that if something is translated as "" it will appear to most tools as if it is translated.

#### brackets (\*args, \*\*kwargs)

Checks that the number of brackets in both strings match.

If ([{ or }]) appear in the original this will check that the same number appear in the translation.

## checker\_name

Extract checker name, for example 'mozilla' from MozillaChecker.

## compendiumconflicts(\*args, \*\*kwargs)

Checks for Gettext compendium conflicts (#-#-#-#).

When you use msgcat to create a PO compendium it will insert #-#-#-# into entries that are not consistent. If the compendium is used later in a message merge then these conflicts will appear in your translations. This test quickly extracts those for correction.

```
credits(*args, **kwargs)
```

Checks for messages containing translation credits instead of normal translations.

Some projects have consistent ways of giving credit to translators by having a unit or two where translators can fill in their name and possibly their contact details. This test allows you to find these units easily to check that they are completed correctly and also disables other tests that might incorrectly get triggered for these units (such as urls, emails, etc.)

## doublequoting(\*args, \*\*kwargs)

Checks whether doublequoting is consistent between the two strings.

Checks on double quotes " to ensure that you have the same number in both the original and the translated string. This tests takes into account that several languages use different quoting characters, and will test for them instead.

## doublespacing(\*args, \*\*kwargs)

Checks for bad double-spaces by comparing to original.

This will identify if you have [space][space] in when you don't have it in the original or it appears in the original but not in your translation. Some of these are spurious and how you correct them depends on the conventions of your language.

### doublewords (\*args, \*\*kwargs)

Checks for repeated words in the translation.

Words that have been repeated in a translation will be highlighted with this test e.g. "the the", "a a". These are generally typos that need correcting. Some languages may have valid repeated words in their structure, in that case either ignore those instances or switch this test off.

### emails (\*args, \*\*kwargs)

Checks that emails are not translated.

Generally you should not be translating email addresses. This check will look to see that email addresses e.g. info@example.com are not translated. In some cases of course you should translate the address but generally you shouldn't.

# endpunc (\*args, \*\*kwargs)

Checks whether punctuation at the end of the strings match.

This will ensure that the ending of your translation has the same punctuation as the original. E.g. if it ends in :[space] then so should yours. It is useful for ensuring that you have ellipses [...] in all your translations, not simply three separate full-stops. You may pick up some errors in the original: feel free to keep your translation and notify the programmers. In some languages, characters such as ? or ! are always preceded by a space e.g. [space]? — do what your language customs dictate. Other false positives you will notice are, for example, if through changes in word-order you add "), etc. at the end of the sentence. Do not change these: your language word-order takes precedence.

It must be noted that if you are tempted to leave out [full-stop] or [colon] or add [full-stop] to a sentence, that often these have been done for a reason, e.g. a list where fullstops make it look cluttered. So, initially match them with the English, and make changes once the program is being used.

This check is aware of several language conventions for punctuation characters, such as the custom question marks for Greek and Arabic, Devanagari Danda, full-width punctuation for CJK languages, etc. Support for your language can be added easily if it is not there yet.

### endwhitespace(\*args, \*\*kwargs)

Checks whether whitespace at the end of the strings matches.

Operates the same as endpunc but is only concerned with whitespace. This filter is particularly useful for those strings which will evidently be followed by another string in the program, e.g. [Password: ] or [Enter your username: ]. The whitespace is an inherent part of the string. This filter makes sure you don't miss those important but otherwise invisible spaces!

If your language uses full-width punctuation (like Chinese), the visual spacing in the character might be enough without an added extra space.

### escapes (\*args, \*\*kwargs)

Checks whether escaping is consistent between the two strings.

Checks escapes such as  $\n \u0000$  to ensure that if they exist in the original string you also have them in the translation.

#### filepaths (\*args, \*\*kwargs)

Checks that file paths have not been translated.

Checks that paths such as /home/user1 have not been translated. Generally you do not translate a file path, unless it is being used as an example, e.g. your\_user\_name/path/to/filename.conf.

### filteraccelerators\_by\_list(str1, acceptlist=None)

Filter out accelerators from str1.

### functions (\*args, \*\*kwargs)

Checks that function names are not translated.

Checks that function names e.g. rgb() or getEntity.Name() are not translated.

### get\_ignored\_filters()

Return checker's additional filters for current language.

### getfilters (excludefilters=None, limitfilters=None)

Returns dictionary of available filters, including/excluding those in the given lists.

### kdecomments (\*args, \*\*kwargs)

Checks to ensure that no KDE style comments appear in the translation.

KDE style translator comments appear in PO files as "\_: comment\n". New translators often translate the comment. This test tries to identify instances where the comment has been translated.

### long (\*args, \*\*kwargs)

Checks whether a translation is much longer than the original string.

This is most useful in the special case where the translation is multiple characters long while the source text is only 1 character long. Otherwise, we use a general ratio that will catch very big differences but is set conservatively to limit the number of false positives.

### musttranslatewords(\*args, \*\*kwargs)

Checks that words configured as definitely translatable don't appear in the translation.

If for instance in your language you decide that you must translate 'OK' then this test will flag any occurrences of 'OK' in the translation if it appeared in the source string. You must specify a file containing all of the *must translate* words using --musttranslatefile.

### newlines (\*args, \*\*kwargs)

Checks whether newlines are consistent between the two strings.

Counts the number of \n newlines (and variants such as \r\n) and reports and error if they differ.

### notranslatewords (\*args, \*\*kwargs)

Checks that words configured as untranslatable appear in the translation too.

Many brand names should not be translated, this test allows you to easily make sure that words like: Word, Excel, Impress, Calc, etc. are not translated. You must specify a file containing all of the *no translate* words using --notranslatefile.

```
numbers (*args, **kwargs)
```

Checks whether numbers of various forms are consistent between the two strings.

You will see some errors where you have either written the number in full or converted it to the digit in your translation. Also changes in order will trigger this error.

```
options (*args, **kwargs)
```

Checks that command line options are not translated.

In messages that contain command line options, such as <code>--help</code>, this test will check that these remain untranslated. These could be translated in the future if programs can create a mechanism to allow this, but currently they are not translated. If the options has a parameter, e.g. <code>--file=FILE</code>, then the test will check that the parameter has been translated.

#### printf(\*args, \*\*kwargs)

Checks whether printf format strings match.

If the printf formatting variables are not identical, then this will indicate an error. Printf statements are used by programs to format output in a human readable form (they are placeholders for variable data). They allow you to specify lengths of string variables, string padding, number padding, precision, etc. Generally

they will look like this: %d, %5.2f, %100s, etc. The test can also manage variables-reordering using the %1\$s syntax. The variables' type and details following data are tested to ensure that they are strictly identical, but they may be reordered.

See also printf Format String.

## puncspacing(\*args, \*\*kwargs)

Checks for bad spacing after punctuation.

In the case of [full-stop][space] in the original, this test checks that your translation does not remove the space. It checks also for [comma], [colon], etc.

Some languages don't use spaces after common punctuation marks, especially where full-width punctuation marks are used. This check will take that into account.

# purepunc (\*args, \*\*kwargs)

Checks that strings that are purely punctuation are not changed.

This extracts strings like + or – as these usually should not be changed.

### pythonbraceformat (\*args, \*\*kwargs)

Checks whether python brace format strings match.

#### run\_filters (unit, categorised=False)

Do some optimisation by caching some data of the unit for the benefit of run\_test().

#### run\_test (test, unit)

Runs the given test on the given unit.

Note that this can raise a FilterFailure as part of normal operation.

## sentencecount (\*args, \*\*kwargs)

Checks that the number of sentences in both strings match.

Adds the number of sentences to see that the sentence count is the same between the original and translated string. You may not always want to use this test, if you find you often need to reformat your translation, because the original is badly-expressed, or because the structure of your language works better that way. Do what works best for your language: it's the meaning of the original you want to convey, not the exact way it was written in the English.

#### setconfig(config)

Sets the accelerator list.

### setsuggestionstore(store)

Sets the filename that a checker should use for evaluating suggestions.

```
short (*args, **kwargs)
```

Checks whether a translation is much shorter than the original string.

This is most useful in the special case where the translation is 1 characters long while the source text is multiple characters long. Otherwise, we use a general ratio that will catch very big differences but is set conservatively to limit the number of false positives.

## simplecaps (\*args, \*\*kwargs)

Checks the capitalisation of two strings isn't wildly different.

This will pick up many false positives, so don't be a slave to it. It is useful for identifying translations that don't start with a capital letter (upper-case letter) when they should, or those that do when they shouldn't. It will also highlight sentences that have extra capitals; depending on the capitalisation convention of your language, you might want to change these to Title Case, or change them all to normal sentence case.

### simpleplurals(\*args, \*\*kwargs)

Checks for English style plural(s) for you to review.

This test will extract any message that contains words with a final "(s)" in the source text. You can then inspect the message, to check that the correct plural form has been used for your language. In some languages, plurals are made by adding text at the beginning of words, making the English style messy. In this case, they often revert to the plural form. This test allows an editor to check that the plurals used are correct. Be aware that this test may create a number of false positives.

For languages with no plural forms (only one noun form) this test will simply test that nothing like "(s)" was used in the translation.

## singlequoting(\*args, \*\*kwargs)

Checks whether singlequoting is consistent between the two strings.

The same as doublequoting but checks for the 'character. Because this is used in contractions like it's and in possessive forms like user's, this test can output spurious errors if your language doesn't use such forms. If a quote appears at the end of a sentence in the translation, i.e. '., this might not be detected properly by the check.

## spellcheck (\*args, \*\*kwargs)

Checks words that don't pass a spell check.

This test will check for misspelled words in your translation. The test first checks for misspelled words in the original (usually English) text, and adds those to an exclusion list. The advantage of this exclusion is that many words that are specific to the application will not raise errors e.g. program names, brand names, function names.

The checker works with PyEnchant. You need to have PyEnchant installed as well as a dictionary for your language (for example, one of the Hunspell or aspell dictionaries). This test will only work if you have specified the --language option.

The pofilter error that is created, lists the misspelled word, plus suggestions returned from the spell checker. That makes it easy for you to identify the word and select a replacement.

## startcaps (\*args, \*\*kwargs)

Checks that the message starts with the correct capitalisation.

After stripping whitespace and common punctuation characters, it then checks to see that the first remaining character is correctly capitalised. So, if the sentence starts with an upper-case letter, and the translation does not, an error is produced.

This check is entirely disabled for many languages that don't make a distinction between upper and lower case. Contact us if this is not yet disabled for your language.

# startpunc(\*args, \*\*kwargs)

Checks whether punctuation at the beginning of the strings match.

Operates as endpunc but you will probably see fewer errors.

#### startwhitespace (\*args, \*\*kwargs)

Checks whether whitespace at the beginning of the strings matches.

As in endwhitespace but you will see fewer errors.

## tabs (\*args, \*\*kwargs)

Checks whether tabs are consistent between the two strings.

Counts the number of \t tab markers and reports an error if they differ.

### unchanged(\*args, \*\*kwargs)

Checks whether a translation is basically identical to the original string.

This checks to see if the translation isn't just a copy of the English original. Sometimes, this is what you want, but other times you will detect words that should have been translated.

```
untranslated(*args, **kwargs)
```

Checks whether a string has been translated at all.

This check is really only useful if you want to extract untranslated strings so that they can be translated independently of the main work.

```
urls (*args, **kwargs)
```

Checks that URLs are not translated.

This checks only basic URLs (http, ftp, mailto etc.) not all URIs (e.g. afp, smb, file). Generally, you don't want to translate URLs, unless they are example URLs (http://your\_server.com/filename.html). If the URL is for configuration information, then you need to query the developers about placing configuration information in PO files. It shouldn't really be there, unless it is very clearly marked: such information should go into a configuration file.

```
validchars (*args, **kwargs)
```

Checks that only characters specified as valid appear in the translation.

Often during character conversion to and from UTF-8 you get some strange characters appearing in your translation. This test presents a simple way to try and identify such errors.

This test will only run of you specify the --validcharsfile command line option. This file contains all the characters that are valid in your language. You must use UTF-8 encoding for the characters in the file.

If the test finds any characters not in your valid characters file then the test will print the character together with its Unicode value (e.g. 002B).

```
variables (*args, **kwargs)
```

Checks whether variables of various forms are consistent between the two strings.

This checks to make sure that variables that appear in the original also appear in the translation. It can handle variables from projects like KDE or OpenOffice. It does not at the moment cope with variables that use the reordering syntax of Gettext PO files.

```
xmltags (*args, **kwargs)
```

Checks that XML/HTML tags have not been translated.

This check finds the number of tags in the source string and checks that the same number are in the translation. If the counts don't match then either the tag is missing or it was mistakenly translated by the translator, both of which are errors.

The check ignores tags or things that look like tags that cover the whole string e.g. <Error> but will produce false positives for things like An <Error> occurred as here Error should be translated. It also will allow translation of the *alt* attribute in e.g. <img src="bob.png" alt="Image description"> or similar translatable attributes in OpenOffice.org help files.

```
exception translate.filters.checks.SeriousFilterFailure(messages)
```

This exception signals that a Filter didn't pass, and the bad translation might break an application (so the string will be marked fuzzy)

The basic test suite for source -> target translations.

```
accelerators (*args, **kwargs)
```

Checks whether accelerators are consistent between the two strings.

This test is capable of checking the different type of accelerators that are used in different projects, like Mozilla or KDE. The test will pick up accelerators that are missing and ones that shouldn't be there.

See accelerators on the localization guide for a full description on accelerators.

#### acronyms (\*args, \*\*kwargs)

Checks that acronyms that appear are unchanged.

If an acronym appears in the original this test will check that it appears in the translation. Translating acronyms is a language decision but many languages leave them unchanged. In that case this test is useful for tracking down translations of the acronym and correcting them.

#### blank (\*args, \*\*kwargs)

Checks whether a translation is totally blank.

This will check to see if a translation has inadvertently been translated as blank i.e. as spaces. This is different from untranslated which is completely empty. This test is useful in that if something is translated as "" it will appear to most tools as if it is translated.

### brackets (\*args, \*\*kwargs)

Checks that the number of brackets in both strings match.

If ([{ or }]) appear in the original this will check that the same number appear in the translation.

#### checker\_name

Extract checker name, for example 'mozilla' from MozillaChecker.

#### compendiumconflicts(\*args, \*\*kwargs)

Checks for Gettext compendium conflicts (#-#-#-#).

When you use msgcat to create a PO compendium it will insert #-#-#-# into entries that are not consistent. If the compendium is used later in a message merge then these conflicts will appear in your translations. This test quickly extracts those for correction.

#### credits (\*args, \*\*kwargs)

Checks for messages containing translation credits instead of normal translations.

Some projects have consistent ways of giving credit to translators by having a unit or two where translators can fill in their name and possibly their contact details. This test allows you to find these units easily to check that they are completed correctly and also disables other tests that might incorrectly get triggered for these units (such as urls, emails, etc.)

### doublequoting(\*args, \*\*kwargs)

Checks whether doublequoting is consistent between the two strings.

Checks on double quotes " to ensure that you have the same number in both the original and the translated string. This tests takes into account that several languages use different quoting characters, and will test for them instead.

### doublespacing(\*args, \*\*kwargs)

Checks for bad double-spaces by comparing to original.

This will identify if you have [space][space] in when you don't have it in the original or it appears in the original but not in your translation. Some of these are spurious and how you correct them depends on the conventions of your language.

## doublewords (\*args, \*\*kwargs)

Checks for repeated words in the translation.

Words that have been repeated in a translation will be highlighted with this test e.g. "the the", "a a". These are generally typos that need correcting. Some languages may have valid repeated words in their structure, in that case either ignore those instances or switch this test off.

## emails (\*args, \*\*kwargs)

Checks that emails are not translated.

Generally you should not be translating email addresses. This check will look to see that email addresses e.g. info@example.com are not translated. In some cases of course you should translate the address but generally you shouldn't.

### endpunc (\*args, \*\*kwargs)

Checks whether punctuation at the end of the strings match.

This will ensure that the ending of your translation has the same punctuation as the original. E.g. if it ends in :[space] then so should yours. It is useful for ensuring that you have ellipses [...] in all your translations, not simply three separate full-stops. You may pick up some errors in the original: feel free to keep your translation and notify the programmers. In some languages, characters such as ? or ! are always preceded by a space e.g. [space]? — do what your language customs dictate. Other false positives you will notice are, for example, if through changes in word-order you add "), etc. at the end of the sentence. Do not change these: your language word-order takes precedence.

It must be noted that if you are tempted to leave out [full-stop] or [colon] or add [full-stop] to a sentence, that often these have been done for a reason, e.g. a list where fullstops make it look cluttered. So, initially match them with the English, and make changes once the program is being used.

This check is aware of several language conventions for punctuation characters, such as the custom question marks for Greek and Arabic, Devanagari Danda, full-width punctuation for CJK languages, etc. Support for your language can be added easily if it is not there yet.

# endwhitespace(\*args, \*\*kwargs)

Checks whether whitespace at the end of the strings matches.

Operates the same as endpunc but is only concerned with whitespace. This filter is particularly useful for those strings which will evidently be followed by another string in the program, e.g. [Password: ] or [Enter your username: ]. The whitespace is an inherent part of the string. This filter makes sure you don't miss those important but otherwise invisible spaces!

If your language uses full-width punctuation (like Chinese), the visual spacing in the character might be enough without an added extra space.

## escapes (\*args, \*\*kwargs)

Checks whether escaping is consistent between the two strings.

Checks escapes such as  $\n \u0000$  to ensure that if they exist in the original string you also have them in the translation.

## filepaths (\*args, \*\*kwargs)

Checks that file paths have not been translated.

Checks that paths such as /home/user1 have not been translated. Generally you do not translate a file path, unless it is being used as an example, e.g. your\_user\_name/path/to/filename.conf.

#### filteraccelerators by list(strl, acceptlist=None)

Filter out accelerators from str1.

## functions (\*args, \*\*kwargs)

Checks that function names are not translated.

Checks that function names e.g. rgb() or getEntity.Name() are not translated.

#### get\_ignored\_filters()

Return checker's additional filters for current language.

## getfilters (excludefilters=None, limitfilters=None)

Returns dictionary of available filters, including/excluding those in the given lists.

### kdecomments (\*args, \*\*kwargs)

Checks to ensure that no KDE style comments appear in the translation.

KDE style translator comments appear in PO files as "\_: comment\n". New translators often translate the comment. This test tries to identify instances where the comment has been translated.

### long (\*args, \*\*kwargs)

Checks whether a translation is much longer than the original string.

This is most useful in the special case where the translation is multiple characters long while the source text is only 1 character long. Otherwise, we use a general ratio that will catch very big differences but is set conservatively to limit the number of false positives.

## musttranslatewords(\*args, \*\*kwargs)

Checks that words configured as definitely translatable don't appear in the translation.

If for instance in your language you decide that you must translate 'OK' then this test will flag any occurrences of 'OK' in the translation if it appeared in the source string. You must specify a file containing all of the *must translate* words using --musttranslatefile.

### newlines (\*args, \*\*kwargs)

Checks whether newlines are consistent between the two strings.

Counts the number of \n newlines (and variants such as \r\n) and reports and error if they differ.

#### notranslatewords (\*args, \*\*kwargs)

Checks that words configured as untranslatable appear in the translation too.

Many brand names should not be translated, this test allows you to easily make sure that words like: Word, Excel, Impress, Calc, etc. are not translated. You must specify a file containing all of the *no translate* words using --notranslatefile.

#### numbers (\*args, \*\*kwargs)

Checks whether numbers of various forms are consistent between the two strings.

You will see some errors where you have either written the number in full or converted it to the digit in your translation. Also changes in order will trigger this error.

### options (\*args, \*\*kwargs)

Checks that command line options are not translated.

In messages that contain command line options, such as <code>--help</code>, this test will check that these remain untranslated. These could be translated in the future if programs can create a mechanism to allow this, but currently they are not translated. If the options has a parameter, e.g. <code>--file=FILE</code>, then the test will check that the parameter has been translated.

# printf(\*args, \*\*kwargs)

Checks whether printf format strings match.

If the printf formatting variables are not identical, then this will indicate an error. Printf statements are used by programs to format output in a human readable form (they are placeholders for variable data). They allow you to specify lengths of string variables, string padding, number padding, precision, etc. Generally they will look like this: %d, %5.2f, %100s, etc. The test can also manage variables-reordering using the %1\$s syntax. The variables' type and details following data are tested to ensure that they are strictly identical, but they may be reordered.

See also printf Format String.

#### puncspacing(\*args, \*\*kwargs)

Checks for bad spacing after punctuation.

In the case of [full-stop][space] in the original, this test checks that your translation does not remove the space. It checks also for [comma], [colon], etc.

Some languages don't use spaces after common punctuation marks, especially where full-width punctuation marks are used. This check will take that into account.

#### purepunc (\*args, \*\*kwargs)

Checks that strings that are purely punctuation are not changed.

This extracts strings like + or - as these usually should not be changed.

### pythonbraceformat (\*args, \*\*kwargs)

Checks whether python brace format strings match.

#### run\_filters (unit, categorised=False)

Do some optimisation by caching some data of the unit for the benefit of run test ().

#### run\_test (test, unit)

Runs the given test on the given unit.

Note that this can raise a *FilterFailure* as part of normal operation.

#### sentencecount (\*args, \*\*kwargs)

Checks that the number of sentences in both strings match.

Adds the number of sentences to see that the sentence count is the same between the original and translated string. You may not always want to use this test, if you find you often need to reformat your translation, because the original is badly-expressed, or because the structure of your language works better that way. Do what works best for your language: it's the meaning of the original you want to convey, not the exact way it was written in the English.

#### setconfig(config)

Sets the accelerator list.

#### setsuggestionstore(store)

Sets the filename that a checker should use for evaluating suggestions.

```
short (*args, **kwargs)
```

Checks whether a translation is much shorter than the original string.

This is most useful in the special case where the translation is 1 characters long while the source text is multiple characters long. Otherwise, we use a general ratio that will catch very big differences but is set conservatively to limit the number of false positives.

```
simplecaps (*args, **kwargs)
```

Checks the capitalisation of two strings isn't wildly different.

This will pick up many false positives, so don't be a slave to it. It is useful for identifying translations that don't start with a capital letter (upper-case letter) when they should, or those that do when they shouldn't. It will also highlight sentences that have extra capitals; depending on the capitalisation convention of your language, you might want to change these to Title Case, or change them all to normal sentence case.

```
simpleplurals(*args, **kwargs)
```

Checks for English style plural(s) for you to review.

This test will extract any message that contains words with a final "(s)" in the source text. You can then inspect the message, to check that the correct plural form has been used for your language. In some languages, plurals are made by adding text at the beginning of words, making the English style messy. In this case, they often revert to the plural form. This test allows an editor to check that the plurals used are correct. Be aware that this test may create a number of false positives.

For languages with no plural forms (only one noun form) this test will simply test that nothing like "(s)" was used in the translation.

## singlequoting(\*args, \*\*kwargs)

Checks whether singlequoting is consistent between the two strings.

The same as doublequoting but checks for the 'character. Because this is used in contractions like it's and in possessive forms like user's, this test can output spurious errors if your language doesn't use such

forms. If a quote appears at the end of a sentence in the translation, i.e. '., this might not be detected properly by the check.

### spellcheck (\*args, \*\*kwargs)

Checks words that don't pass a spell check.

This test will check for misspelled words in your translation. The test first checks for misspelled words in the original (usually English) text, and adds those to an exclusion list. The advantage of this exclusion is that many words that are specific to the application will not raise errors e.g. program names, brand names, function names.

The checker works with PyEnchant. You need to have PyEnchant installed as well as a dictionary for your language (for example, one of the Hunspell or aspell dictionaries). This test will only work if you have specified the --language option.

The pofilter error that is created, lists the misspelled word, plus suggestions returned from the spell checker. That makes it easy for you to identify the word and select a replacement.

## startcaps (\*args, \*\*kwargs)

Checks that the message starts with the correct capitalisation.

After stripping whitespace and common punctuation characters, it then checks to see that the first remaining character is correctly capitalised. So, if the sentence starts with an upper-case letter, and the translation does not, an error is produced.

This check is entirely disabled for many languages that don't make a distinction between upper and lower case. Contact us if this is not yet disabled for your language.

## startpunc(\*args, \*\*kwargs)

Checks whether punctuation at the beginning of the strings match.

Operates as endpunc but you will probably see fewer errors.

## startwhitespace(\*args, \*\*kwargs)

Checks whether whitespace at the beginning of the strings matches.

As in endwhitespace but you will see fewer errors.

### tabs (\*args, \*\*kwargs)

Checks whether tabs are consistent between the two strings.

Counts the number of \t tab markers and reports an error if they differ.

### unchanged (\*args, \*\*kwargs)

Checks whether a translation is basically identical to the original string.

This checks to see if the translation isn't just a copy of the English original. Sometimes, this is what you want, but other times you will detect words that should have been translated.

### untranslated(\*args, \*\*kwargs)

Checks whether a string has been translated at all.

This check is really only useful if you want to extract untranslated strings so that they can be translated independently of the main work.

### urls (\*args, \*\*kwargs)

Checks that URLs are not translated.

This checks only basic URLs (http, ftp, mailto etc.) not all URIs (e.g. afp, smb, file). Generally, you don't want to translate URLs, unless they are example URLs (http://your\_server.com/filename.html). If the URL is for configuration information, then you need to query the developers about placing configuration information in PO files. It shouldn't really be there, unless it is very clearly marked: such information should go into a configuration file.

#### validchars (\*args, \*\*kwargs)

Checks that only characters specified as valid appear in the translation.

Often during character conversion to and from UTF-8 you get some strange characters appearing in your translation. This test presents a simple way to try and identify such errors.

This test will only run of you specify the --validcharsfile command line option. This file contains all the characters that are valid in your language. You must use UTF-8 encoding for the characters in the file.

If the test finds any characters not in your valid characters file then the test will print the character together with its Unicode value (e.g. 002B).

# variables (\*args, \*\*kwargs)

Checks whether variables of various forms are consistent between the two strings.

This checks to make sure that variables that appear in the original also appear in the translation. It can handle variables from projects like KDE or OpenOffice. It does not at the moment cope with variables that use the reordering syntax of Gettext PO files.

```
xmltags (*args, **kwargs)
```

Checks that XML/HTML tags have not been translated.

This check finds the number of tags in the source string and checks that the same number are in the translation. If the counts don't match then either the tag is missing or it was mistakenly translated by the translator, both of which are errors.

The check ignores tags or things that look like tags that cover the whole string e.g. <Error> but will produce false positives for things like An <Error> occurred as here Error should be translated. It also will allow translation of the *alt* attribute in e.g. <img src="bob.png" alt="Image description"> or similar translatable attributes in OpenOffice.org help files.

```
 \textbf{class} \text{ translate.filters.checks.} \textbf{StandardUnitChecker} (\textit{checkerconfig=None}, \quad \textit{exclude-filters=None}, \quad \textit{limitfilters=None}, \\ \textit{errorhandler=None})
```

The standard checks for common checks on translation units.

#### checker name

Extract checker name, for example 'mozilla' from MozillaChecker.

### filteraccelerators\_by\_list(str1, acceptlist=None)

Filter out accelerators from str1.

## get\_ignored\_filters()

Return checker's additional filters for current language.

### getfilters (excludefilters=None, limitfilters=None)

Returns dictionary of available filters, including/excluding those in the given lists.

### hassuggestion(\*args, \*\*kwargs)

Checks if there is at least one suggested translation for this unit.

If a message has a suggestion (an alternate translation stored in alt-trans units in XLIFF and .pending files in PO) then these will be extracted. This is used by Pootle and is probably only useful in pofilter when using XLIFF files.

```
isfuzzy(*args, **kwargs)
```

Check if the unit has been marked fuzzy.

If a message is marked fuzzy in the PO file then it is extracted. Note this is different from --fuzzy and --nofuzzy options which specify whether tests should be performed against messages marked fuzzy.

```
isreview(*args, **kwargs)
```

Check if the unit has been marked review.

If you have made use of the 'review' flags in your translations:

```
# (review) reason for review
# (pofilter) testname: explanation for translator
```

Then if a message is marked for review in the PO file it will be extracted. Note this is different from --review and --noreview options which specify whether tests should be performed against messages already marked as under review.

```
nplurals (*args, **kwargs)
```

Checks for the correct number of noun forms for plural translations.

This uses the plural information in the language module of the Translate Toolkit. This is the same as the Gettext nplural value. It will check that the number of plurals required is the same as the number supplied in your translation.

## run\_filters (unit, categorised=False)

Run all the tests in this suite.

### **Return type** Dictionary

#### **Returns**

Content of the dictionary is as follows:

```
{'testname': { 'message': message_or_exception, 'category': failure_

category } }
```

#### run\_test (test, unit)

Runs the given test on the given unit.

Note that this can raise a FilterFailure as part of normal operation.

# setconfig(config)

Sets the accelerator list.

#### setsuggestionstore(store)

Sets the filename that a checker should use for evaluating suggestions.

A Checker that controls multiple checkers.

```
categories = {}
```

Categories where each checking function falls into Function names are used as keys, categories are the values

```
getfilters (excludefilters=None, limitfilters=None)
```

Returns a dictionary of available filters, including/excluding those in the given lists.

```
run_filters (unit, categorised=False)
```

Run all the tests in the checker's suites.

## setsuggestionstore(store)

Sets the filename that a checker should use for evaluating suggestions.

```
class translate.filters.checks.TermChecker(**kwargs)
```

#### accelerators (\*args, \*\*kwargs)

Checks whether accelerators are consistent between the two strings.

This test is capable of checking the different type of accelerators that are used in different projects, like Mozilla or KDE. The test will pick up accelerators that are missing and ones that shouldn't be there.

See accelerators on the localization guide for a full description on accelerators.

## acronyms (\*args, \*\*kwargs)

Checks that acronyms that appear are unchanged.

If an acronym appears in the original this test will check that it appears in the translation. Translating acronyms is a language decision but many languages leave them unchanged. In that case this test is useful for tracking down translations of the acronym and correcting them.

### blank (\*args, \*\*kwargs)

Checks whether a translation is totally blank.

This will check to see if a translation has inadvertently been translated as blank i.e. as spaces. This is different from untranslated which is completely empty. This test is useful in that if something is translated as "" it will appear to most tools as if it is translated.

### brackets (\*args, \*\*kwargs)

Checks that the number of brackets in both strings match.

If ([{ or }]) appear in the original this will check that the same number appear in the translation.

#### checker name

Extract checker name, for example 'mozilla' from MozillaChecker.

#### compendiumconflicts(\*args, \*\*kwargs)

Checks for Gettext compendium conflicts (#-#-#-#).

When you use msgcat to create a PO compendium it will insert #-#-#-#-# into entries that are not consistent. If the compendium is used later in a message merge then these conflicts will appear in your translations. This test quickly extracts those for correction.

## credits(\*args, \*\*kwargs)

Checks for messages containing translation credits instead of normal translations.

Some projects have consistent ways of giving credit to translators by having a unit or two where translators can fill in their name and possibly their contact details. This test allows you to find these units easily to check that they are completed correctly and also disables other tests that might incorrectly get triggered for these units (such as urls, emails, etc.)

### doublequoting(\*args, \*\*kwargs)

Checks whether doublequoting is consistent between the two strings.

Checks on double quotes " to ensure that you have the same number in both the original and the translated string. This tests takes into account that several languages use different quoting characters, and will test for them instead.

### doublespacing(\*args, \*\*kwargs)

Checks for bad double-spaces by comparing to original.

This will identify if you have [space][space] in when you don't have it in the original or it appears in the original but not in your translation. Some of these are spurious and how you correct them depends on the conventions of your language.

## doublewords (\*args, \*\*kwargs)

Checks for repeated words in the translation.

Words that have been repeated in a translation will be highlighted with this test e.g. "the the", "a a". These are generally typos that need correcting. Some languages may have valid repeated words in their structure, in that case either ignore those instances or switch this test off.

# emails (\*args, \*\*kwargs)

Checks that emails are not translated.

Generally you should not be translating email addresses. This check will look to see that email addresses e.g. info@example.com are not translated. In some cases of course you should translate the address but generally you shouldn't.

## endpunc (\*args, \*\*kwargs)

Checks whether punctuation at the end of the strings match.

This will ensure that the ending of your translation has the same punctuation as the original. E.g. if it ends in :[space] then so should yours. It is useful for ensuring that you have ellipses [...] in all your translations, not simply three separate full-stops. You may pick up some errors in the original: feel free to keep your translation and notify the programmers. In some languages, characters such as ? or ! are always preceded by a space e.g. [space]? — do what your language customs dictate. Other false positives you will notice are, for example, if through changes in word-order you add "), etc. at the end of the sentence. Do not change these: your language word-order takes precedence.

It must be noted that if you are tempted to leave out [full-stop] or [colon] or add [full-stop] to a sentence, that often these have been done for a reason, e.g. a list where fullstops make it look cluttered. So, initially match them with the English, and make changes once the program is being used.

This check is aware of several language conventions for punctuation characters, such as the custom question marks for Greek and Arabic, Devanagari Danda, full-width punctuation for CJK languages, etc. Support for your language can be added easily if it is not there yet.

### endwhitespace(\*args, \*\*kwargs)

Checks whether whitespace at the end of the strings matches.

Operates the same as endpunc but is only concerned with whitespace. This filter is particularly useful for those strings which will evidently be followed by another string in the program, e.g. [Password: ] or [Enter your username: ]. The whitespace is an inherent part of the string. This filter makes sure you don't miss those important but otherwise invisible spaces!

If your language uses full-width punctuation (like Chinese), the visual spacing in the character might be enough without an added extra space.

### escapes (\*args, \*\*kwargs)

Checks whether escaping is consistent between the two strings.

Checks escapes such as  $\n \u0000$  to ensure that if they exist in the original string you also have them in the translation.

### filepaths (\*args, \*\*kwargs)

Checks that file paths have not been translated.

Checks that paths such as /home/user1 have not been translated. Generally you do not translate a file path, unless it is being used as an example, e.g. your\_user\_name/path/to/filename.conf.

#### filteraccelerators\_by\_list(str1, acceptlist=None)

Filter out accelerators from str1.

### functions (\*args, \*\*kwargs)

Checks that function names are not translated.

Checks that function names e.g. rgb() or getEntity.Name() are not translated.

#### get ignored filters()

Return checker's additional filters for current language.

### getfilters (excludefilters=None, limitfilters=None)

Returns dictionary of available filters, including/excluding those in the given lists.

#### kdecomments (\*args, \*\*kwargs)

Checks to ensure that no KDE style comments appear in the translation.

KDE style translator comments appear in PO files as "\_: comment\n". New translators often translate the comment. This test tries to identify instances where the comment has been translated.

```
long (*args, **kwargs)
```

Checks whether a translation is much longer than the original string.

This is most useful in the special case where the translation is multiple characters long while the source text is only 1 character long. Otherwise, we use a general ratio that will catch very big differences but is set conservatively to limit the number of false positives.

## musttranslatewords(\*args, \*\*kwargs)

Checks that words configured as definitely translatable don't appear in the translation.

If for instance in your language you decide that you must translate 'OK' then this test will flag any occurrences of 'OK' in the translation if it appeared in the source string. You must specify a file containing all of the *must translate* words using --musttranslatefile.

```
newlines (*args, **kwargs)
```

Checks whether newlines are consistent between the two strings.

Counts the number of \n newlines (and variants such as \r\n) and reports and error if they differ.

### notranslatewords (\*args, \*\*kwargs)

Checks that words configured as untranslatable appear in the translation too.

Many brand names should not be translated, this test allows you to easily make sure that words like: Word, Excel, Impress, Calc, etc. are not translated. You must specify a file containing all of the *no translate* words using --notranslatefile.

```
numbers (*args, **kwargs)
```

Checks whether numbers of various forms are consistent between the two strings.

You will see some errors where you have either written the number in full or converted it to the digit in your translation. Also changes in order will trigger this error.

```
options (*args, **kwargs)
```

Checks that command line options are not translated.

In messages that contain command line options, such as --help, this test will check that these remain untranslated. These could be translated in the future if programs can create a mechanism to allow this, but currently they are not translated. If the options has a parameter, e.g. --file=FILE, then the test will check that the parameter has been translated.

```
printf(*args, **kwargs)
```

Checks whether printf format strings match.

If the printf formatting variables are not identical, then this will indicate an error. Printf statements are used by programs to format output in a human readable form (they are placeholders for variable data). They allow you to specify lengths of string variables, string padding, number padding, precision, etc. Generally they will look like this: %d, %5.2f, %100s, etc. The test can also manage variables-reordering using the %1\$s syntax. The variables' type and details following data are tested to ensure that they are strictly identical, but they may be reordered.

See also printf Format String.

#### puncspacing(\*args, \*\*kwargs)

Checks for bad spacing after punctuation.

In the case of [full-stop][space] in the original, this test checks that your translation does not remove the space. It checks also for [comma], [colon], etc.

Some languages don't use spaces after common punctuation marks, especially where full-width punctuation marks are used. This check will take that into account.

#### purepunc (\*args, \*\*kwargs)

Checks that strings that are purely punctuation are not changed.

This extracts strings like + or - as these usually should not be changed.

### pythonbraceformat (\*args, \*\*kwargs)

Checks whether python brace format strings match.

## run\_filters (unit, categorised=False)

Do some optimisation by caching some data of the unit for the benefit of run\_test().

#### run\_test (test, unit)

Runs the given test on the given unit.

Note that this can raise a FilterFailure as part of normal operation.

### sentencecount (\*args, \*\*kwargs)

Checks that the number of sentences in both strings match.

Adds the number of sentences to see that the sentence count is the same between the original and translated string. You may not always want to use this test, if you find you often need to reformat your translation, because the original is badly-expressed, or because the structure of your language works better that way. Do what works best for your language: it's the meaning of the original you want to convey, not the exact way it was written in the English.

## setconfig(config)

Sets the accelerator list.

## setsuggestionstore(store)

Sets the filename that a checker should use for evaluating suggestions.

#### short (\*args, \*\*kwargs)

Checks whether a translation is much shorter than the original string.

This is most useful in the special case where the translation is 1 characters long while the source text is multiple characters long. Otherwise, we use a general ratio that will catch very big differences but is set conservatively to limit the number of false positives.

#### simplecaps (\*args, \*\*kwargs)

Checks the capitalisation of two strings isn't wildly different.

This will pick up many false positives, so don't be a slave to it. It is useful for identifying translations that don't start with a capital letter (upper-case letter) when they should, or those that do when they shouldn't. It will also highlight sentences that have extra capitals; depending on the capitalisation convention of your language, you might want to change these to Title Case, or change them all to normal sentence case.

#### simpleplurals(\*args, \*\*kwargs)

Checks for English style plural(s) for you to review.

This test will extract any message that contains words with a final "(s)" in the source text. You can then inspect the message, to check that the correct plural form has been used for your language. In some languages, plurals are made by adding text at the beginning of words, making the English style messy. In this case, they often revert to the plural form. This test allows an editor to check that the plurals used are correct. Be aware that this test may create a number of false positives.

For languages with no plural forms (only one noun form) this test will simply test that nothing like "(s)" was used in the translation.

### singlequoting(\*args, \*\*kwargs)

Checks whether singlequoting is consistent between the two strings.

The same as doublequoting but checks for the 'character. Because this is used in contractions like it's and in possessive forms like user's, this test can output spurious errors if your language doesn't use such forms. If a quote appears at the end of a sentence in the translation, i.e. '., this might not be detected properly by the check.

### spellcheck (\*args, \*\*kwargs)

Checks words that don't pass a spell check.

This test will check for misspelled words in your translation. The test first checks for misspelled words in the original (usually English) text, and adds those to an exclusion list. The advantage of this exclusion is that many words that are specific to the application will not raise errors e.g. program names, brand names, function names.

The checker works with PyEnchant. You need to have PyEnchant installed as well as a dictionary for your language (for example, one of the Hunspell or aspell dictionaries). This test will only work if you have specified the --language option.

The pofilter error that is created, lists the misspelled word, plus suggestions returned from the spell checker. That makes it easy for you to identify the word and select a replacement.

### startcaps (\*args, \*\*kwargs)

Checks that the message starts with the correct capitalisation.

After stripping whitespace and common punctuation characters, it then checks to see that the first remaining character is correctly capitalised. So, if the sentence starts with an upper-case letter, and the translation does not, an error is produced.

This check is entirely disabled for many languages that don't make a distinction between upper and lower case. Contact us if this is not yet disabled for your language.

### startpunc(\*args, \*\*kwargs)

Checks whether punctuation at the beginning of the strings match.

Operates as endpunc but you will probably see fewer errors.

## startwhitespace (\*args, \*\*kwargs)

Checks whether whitespace at the beginning of the strings matches.

As in endwhitespace but you will see fewer errors.

## tabs (\*args, \*\*kwargs)

Checks whether tabs are consistent between the two strings.

Counts the number of \t tab markers and reports an error if they differ.

## unchanged(\*args, \*\*kwargs)

Checks whether a translation is basically identical to the original string.

This checks to see if the translation isn't just a copy of the English original. Sometimes, this is what you want, but other times you will detect words that should have been translated.

#### untranslated(\*args, \*\*kwargs)

Checks whether a string has been translated at all.

This check is really only useful if you want to extract untranslated strings so that they can be translated independently of the main work.

```
urls (*args, **kwargs)
```

Checks that URLs are not translated.

This checks only basic URLs (http, ftp, mailto etc.) not all URIs (e.g. afp, smb, file). Generally, you don't want to translate URLs, unless they are example URLs (http://your\_server.com/filename.html). If the URL is for configuration information, then you need to query the developers about placing configuration information in PO files. It shouldn't really be there, unless it is very clearly marked: such information should go into a configuration file.

#### validchars (\*args, \*\*kwargs)

Checks that only characters specified as valid appear in the translation.

Often during character conversion to and from UTF-8 you get some strange characters appearing in your translation. This test presents a simple way to try and identify such errors.

This test will only run of you specify the --validcharsfile command line option. This file contains all the characters that are valid in your language. You must use UTF-8 encoding for the characters in the file.

If the test finds any characters not in your valid characters file then the test will print the character together with its Unicode value (e.g. 002B).

```
variables (*args, **kwargs)
```

Checks whether variables of various forms are consistent between the two strings.

This checks to make sure that variables that appear in the original also appear in the translation. It can handle variables from projects like KDE or OpenOffice. It does not at the moment cope with variables that use the reordering syntax of Gettext PO files.

```
xmltags (*args, **kwargs)
```

Checks that XML/HTML tags have not been translated.

This check finds the number of tags in the source string and checks that the same number are in the translation. If the counts don't match then either the tag is missing or it was mistakenly translated by the translator, both of which are errors.

The check ignores tags or things that look like tags that cover the whole string e.g. <Error> but will produce false positives for things like An <Error> occurred as here Error should be translated. It also will allow translation of the *alt* attribute in e.g. <img src="bob.png" alt="Image description"> or similar translatable attributes in OpenOffice.org help files.

A checker that passes source and target strings to the checks, not the whole unit.

This provides some speedup and simplifies testing.

### checker\_name

Extract checker name, for example 'mozilla' from MozillaChecker.

```
filteraccelerators_by_list(str1, acceptlist=None)
```

Filter out accelerators from str1.

### get\_ignored\_filters()

Return checker's additional filters for current language.

```
getfilters (excludefilters=None, limitfilters=None)
```

Returns dictionary of available filters, including/excluding those in the given lists.

```
run filters (unit, categorised=False)
```

Do some optimisation by caching some data of the unit for the benefit of run\_test().

```
run test(test, unit)
```

Runs the given test on the given unit.

Note that this can raise a FilterFailure as part of normal operation.

### setconfig(config)

Sets the accelerator list.

#### setsuggestionstore(store)

Sets the filename that a checker should use for evaluating suggestions.

Parent Checker class which does the checking based on functions available in derived classes.

## categories = None

Categories where each checking function falls into Function names are used as keys, categories are the values

## checker\_name

Extract checker name, for example 'mozilla' from MozillaChecker.

```
filteraccelerators_by_list(str1, acceptlist=None)
```

Filter out accelerators from str1.

### get\_ignored\_filters()

Return checker's additional filters for current language.

```
getfilters (excludefilters=None, limitfilters=None)
```

Returns dictionary of available filters, including/excluding those in the given lists.

```
run_filters (unit, categorised=False)
```

Run all the tests in this suite.

## Return type Dictionary

## Returns

Content of the dictionary is as follows:

```
{'testname': { 'message': message_or_exception, 'category': failure_

-category } }
```

### run\_test (test, unit)

Runs the given test on the given unit.

Note that this can raise a *FilterFailure* as part of normal operation.

#### setconfig(config)

Sets the accelerator list.

### setsuggestionstore(store)

Sets the filename that a checker should use for evaluating suggestions.

```
translate.filters.checks.batchruntests(pairs)
```

Runs test on a batch of string pairs.

```
translate.filters.checks.intuplelist (pair, list)
```

Tests to see if pair == (a,b,c) is in list, but handles None entries in list as wildcards (only allowed in positions "a" and "c"). We take a shortcut by only considering "c" if "b" has already matched.

```
translate.filters.checks.runtests(str1, str2, ignorelist=())
```

Verifies that the tests pass for a pair of strings.

```
translate.filters.checks.tagname(string)
     Returns the name of the XML/HTML tag in string
translate.filters.checks.tagproperties(strings, ignore)
     Returns all the properties in the XML/HTML tag string as (tagname, propertyname, propertyvalue), but ignore
     those combinations specified in ignore.
decoration
functions to get decorative/informative text out of strings...
translate.filters.decoration.countaccelerators(accelmarker, acceptlist=None)
     returns a function that counts the number of accelerators marked with the given marker
translate.filters.decoration.findaccelerators(str1, accelmarker, acceptlist=None)
     returns all the accelerators and locations in str1 marked with a given marker
translate.filters.decoration.findmarkedvariables(strl, startmarker, endmarker, ig-
                                                                norelist=[]
     returns all the variables and locations in str1 marked with a given marker
translate.filters.decoration.getaccelerators(accelmarker, acceptlist=None)
     returns a function that gets a list of accelerators marked using accelmarker
translate.filters.decoration.getemails(str1)
     returns the email addresses that are in a string
translate.filters.decoration.getfunctions(str1)
     returns the functions() that are in a string, while ignoring the trailing punctuation in the given parameter
translate.filters.decoration.getnumbers(str1)
     returns any numbers that are in the string
translate.filters.decoration.geturls(str1)
     returns the URIs in a string
translate.filters.decoration.qetvariables(startmarker, endmarker)
     returns a function that gets a list of variables marked using startmarker and endmarker
translate.filters.decoration.ispurepunctuation(str1)
     checks whether the string is entirely punctuation
translate.filters.decoration.isvalidaccelerator(accelerator, acceptlist=None)
     returns whether the given accelerator character is valid
          Parameters
                • accelerator (character) - A character to be checked for accelerator validity
                • acceptlist (String) – A list of characters that are permissible as accelerators
          Return type Boolean
          Returns True if the supplied character is an acceptable accelerator
translate.filters.decoration.puncend(str1, punctuation)
     returns all the punctuation from the end of the string
translate.filters.decoration.puncstart(str1, punctuation)
     returns all the punctuation from the start of the string
translate.filters.decoration.spaceend(str1)
     returns all the whitespace from the end of the string
```

```
translate.filters.decoration.spacestart (strI) returns all the whitespace from the start of the string
```

### helpers

```
a set of helper functions for filters...
translate.filters.helpers.countmatch(str1, str2, countstr)
     checks whether countstr occurs the same number of times in str1 and str2
translate.filters.helpers.countsmatch(str1, str2, countlist)
     checks whether each element in countlist occurs the same number of times in str1 and str2
translate.filters.helpers.filtercount (str1, func)
     returns the number of characters in str1 that pass func
translate.filters.helpers.filtertestmethod(testmethod, strfilter)
     returns a version of the testmethod that operates on filtered strings using strfilter
translate.filters.helpers.funcmatch(str1, str2, func, *args)
     returns whether the result of func is the same for str1 and str2
translate.filters.helpers.funcsmatch(str1, str2, funclist)
     checks whether the results of each func in funclist match for str1 and str2
translate.filters.helpers.multifilter(strl, strfilters, *args)
     passes str1 through a list of filters
translate.filters.helpers.multifiltertestmethod(testmethod, strfilters)
     returns a version of the testmethod that operates on filtered strings using strfilter
```

#### pofilter

Perform quality checks on Gettext PO, XLIFF and TMX localization files.

Snippet files are created whenever a test fails. These can be examined, corrected and merged back into the originals using pomerge.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/pofilter.html for examples and usage instructions and http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/pofilter\_tests.html for full descriptions of all tests.

```
class translate.filters.pofilter.FilterOptionParser (formats)
    A specialized Option Parser for filter tools...

add_option (Option)
        add_option(opt_str,..., kwarg=val,...)

check_values (values: Values, args: [string])
        -> (values: Values, args: [string])
```

Check that the supplied option values and leftover arguments are valid. Returns the option values and leftover arguments (possibly adjusted, possibly completely new – whatever you like). Default implementation just returns the passed-in values; subclasses may override as desired.

```
checkoutputsubdir(options, subdir)
```

Checks to see if subdir under options.output needs to be created, creates if neccessary.

```
define\_option(option)
```

Defines the given option, replacing an existing one of the same short name if neccessary...

#### destroy()

Declare that you are done with this OptionParser. This cleans up reference cycles so the OptionParser (and all objects referenced by it) can be garbage-collected promptly. After calling destroy(), the OptionParser is unusable.

### disable\_interspersed\_args()

Set parsing to stop on the first non-option. Use this if you have a command processor which runs another command that has options of its own and you want to make sure these options don't get confused.

### enable\_interspersed\_args()

Set parsing to not stop on the first non-option, allowing interspersing switches with command arguments. This is the default behavior. See also disable\_interspersed\_args() and the class documentation description of the attribute allow\_interspersed\_args.

#### error (msg : string)

Print a usage message incorporating 'msg' to stderr and exit. If you override this in a subclass, it should not return – it should either exit or raise an exception.

### finalizetempoutputfile (options, outputfile, fulloutputpath)

Write the temp outputfile to its final destination.

#### format\_manpage()

returns a formatted manpage

## getformathelp (formats)

Make a nice help string for describing formats...

### getfullinputpath (options, inputpath)

Gets the absolute path to an input file.

### getfulloutputpath (options, outputpath)

Gets the absolute path to an output file.

## getfulltemplatepath (options, templatepath)

Gets the absolute path to a template file.

# getoutputname (options, inputname, outputformat)

Gets an output filename based on the input filename.

#### getoutputoptions (options, inputpath, templatepath)

Works out which output format and processor method to use...

### getpassthroughoptions (options)

Get the options required to pass to the filtermethod...

### gettemplatename (options, inputname)

Gets an output filename based on the input filename.

#### getusageman (option)

returns the usage string for the given option

## getusagestring(option)

returns the usage string for the given option

#### isexcluded(options, inputpath)

Checks if this path has been excluded.

### isrecursive (fileoption, filepurpose='input')

Checks if fileoption is a recursive file.

## isvalidinputname (inputname)

Checks if this is a valid input filename.

#### mkdir (parent, subdir)

Makes a subdirectory (recursively if neccessary).

### openinputfile (options, fullinputpath)

Opens the input file.

### openoutputfile (options, fulloutputpath)

Opens the output file.

#### opentemplatefile (options, fulltemplatepath)

Opens the template file (if required).

### opentempoutputfile (options, fulloutputpath)

Opens a temporary output file.

#### parse\_args (args=None, values=None)

Parses the command line options, handling implicit input/output args.

#### parse\_noinput (option, opt, value, parser, \*args, \*\*kwargs)

This sets an option to *True*, but also sets input to - to prevent an error.

#### print help(file : file = stdout)

Print an extended help message, listing all options and any help text provided with them, to 'file' (default stdout).

#### print manpage(file=None)

outputs a manpage for the program using the help information

#### print\_usage (file : file = stdout)

Print the usage message for the current program (self.usage) to 'file' (default stdout). Any occurrence of the string "%prog" in self.usage is replaced with the name of the current program (basename of sys.argv[0]). Does nothing if self.usage is empty or not defined.

### print\_version (file : file = stdout)

Print the version message for this program (self.version) to 'file' (default stdout). As with print\_usage(), any occurrence of "%prog" in self.version is replaced by the current program's name. Does nothing if self.version is empty or undefined.

### processfile (fileprocessor, options, fullinputpath, fulloutputpath, fulltemplatepath)

Process an individual file.

# recurseinputfilelist(options)

Use a list of files, and find a common base directory for them.

## recurseinputfiles (options)

Recurse through directories and return files to be processed.

#### recursiveprocess (options)

Recurse through directories and process files.

### run()

Parses the arguments, and runs recursive process with the resulting options.

### set\_usage (usage=None)

sets the usage string - if usage not given, uses getusagestring for each option

#### seterrorleveloptions()

Sets the errorlevel options.

### setformats (formats, usetemplates)

Sets the format options using the given format dictionary.

**Parameters** formats (Dictionary) – The dictionary keys should be:

```
• Single strings (or 1-tuples) containing an input format (if not usetemplates)
```

- Tuples containing an input format and template format (if usetemplates)
- Formats can be *None* to indicate what to do with standard input

The dictionary *values* should be tuples of outputformat (string) and processor method.

```
setmanpageoption()
```

creates a manpage option that allows the optionparser to generate a manpage

## setprogressoptions()

Sets the progress options.

#### splitext (pathname)

Splits *pathname* into name and ext, and removes the extsep.

**Parameters** pathname (string) – A file path

Returns root, ext

Return type tuple

### splitinputext(inputpath)

Splits an *inputpath* into name and extension.

## splittemplateext (templatepath)

Splits a *templatepath* into name and extension.

#### templateexists (options, templatepath)

Returns whether the given template exists...

```
warning (msg, options=None, exc_info=None)
```

Print a warning message incorporating 'msg' to stderr and exit.

```
translate.filters.pofilter.build_checkerconfig(options)
```

Prepare the checker config from the given options. This is mainly factored out for the sake of unit tests.

translate.filters.pofilter.runfilter(inputfile, outputfile, templatefile, checkfilter=None)
Reads in inputfile, filters using checkfilter, writes to outputfile.

### prefilters

Filters that strings can be passed through before certain tests.

```
translate.filters.prefilters.filteraccelerators(accelmarker)
```

Returns a function that filters accelerators marked using accelmarker from a strings.

Parameters accelmarker (string) - Accelerator marker character

**Return type** Function

**Returns** fn(str1, acceplist=None)

translate.filters.prefilters. filtervariables (startmarker, endmarker, varfilter)

Returns a function that filters variables marked using *startmarker* and *endmarker* from a string.

#### **Parameters**

- **startmarker** (*string*) **Start** of variable marker
- endmarker (string) End of variable marker
- varfilter (Function) fn(variable, startmarker, endmarker)

Return type Function

**Returns** fn(str1)

translate.filters.prefilters.filterwordswithpunctuation(str1)

Goes through a list of known words that have punctuation and removes the punctuation from them.

translate.filters.prefilters.removekdecomments(str1)

Remove KDE-style PO comments.

KDE comments start with \_: [space] and end with a literal \n. Example:

"\_: comment\n"

translate.filters.prefilters.varname(variable, startmarker, endmarker)

Variable filter that returns the variable name without the marking punctuation.

Note: Currently this function simply returns variable unchanged, no matter what \*marker's are set to.

Return type String

**Returns** Variable name with the supplied *startmarker* and *endmarker* removed.

translate.filters.prefilters.varnone(variable, startmarker, endmarker)
Variable filter that returns an empty string.

Return type String

**Returns** Empty string

## spelling

An API to provide spell checking for use in checks or elsewhere.

### lang

Classes that represent languages and provides language-specific information.

All classes inherit from the parent class called common.

The type of data includes:

- · Language codes
- Language name
- Plurals
- Punctuation transformation
- etc.

#### af

This module represents the Afrikaans language.

See also:

http://en.wikipedia.org/wiki/Afrikaans\_language

#### class translate.lang.af.af

This class represents Afrikaans.

### classmethod alter\_length(text)

Converts the given string by adding or removing characters as an estimation of translation length (with English assumed as source language).

#### classmethod capsstart(text)

Modify this for the indefinite article ('n).

### classmethod character\_iter(text)

Returns an iterator over the characters in text.

### classmethod characters(text)

Returns a list of characters in text.

#### classmethod length\_difference(length)

Returns an estimate to a likely change in length relative to an English string of length length.

### classmethod numbertranslate(text)

Converts the numbers in a string according to the rules of the language.

## classmethod numstart(text)

Determines whether the text starts with a numeric value.

#### classmethod punctranslate(text)

Converts the punctuation in a string according to the rules of the language.

### classmethod sentence\_iter(text, strip=True)

Returns an iterator over the sentences in text.

### classmethod sentences(text, strip=True)

Returns a list of sentences in text.

#### classmethod word iter(text)

Returns an iterator over the words in text.

### classmethod words (text)

Returns a list of words in text.

```
translate.lang.af.cyr2lat = {u'\u0401': u'Jo', u'\u0410': u'A', u'\u0411': u'B', u'\u041
Mapping of Cyrillic to Latin letters for transliteration in Afrikaans
```

```
translate.lang.af.tranliterate_cyrillic(text)
```

Convert Cyrillic text to Latin according to the AWS transliteration rules.

#### am

This module represents the Amharic language.

#### See also:

http://en.wikipedia.org/wiki/Amharic\_language

#### class translate.lang.am.am

This class represents Amharic.

# classmethod alter\_length(text)

Converts the given string by adding or removing characters as an estimation of translation length (with English assumed as source language).

#### classmethod capsstart(text)

Determines whether the text starts with a capital letter.

### classmethod character\_iter(text)

Returns an iterator over the characters in text.

### classmethod characters(text)

Returns a list of characters in text.

#### classmethod length difference (length)

Returns an estimate to a likely change in length relative to an English string of length length.

### classmethod numbertranslate(text)

Converts the numbers in a string according to the rules of the language.

#### classmethod numstart (text)

Determines whether the text starts with a numeric value.

#### classmethod punctranslate(text)

Converts the punctuation in a string according to the rules of the language.

#### classmethod sentence iter(text, strip=True)

Returns an iterator over the sentences in text.

### classmethod sentences (text, strip=True)

Returns a list of sentences in text.

### classmethod word\_iter(text)

Returns an iterator over the words in text.

#### classmethod words(text)

Returns a list of words in text.

#### ar

This module represents the Arabic language.

### See also:

http://en.wikipedia.org/wiki/Arabic\_language

### class translate.lang.ar.ar

This class represents Arabic.

### classmethod alter\_length(text)

Converts the given string by adding or removing characters as an estimation of translation length (with English assumed as source language).

#### classmethod capsstart (text)

Determines whether the text starts with a capital letter.

### classmethod character\_iter(text)

Returns an iterator over the characters in text.

#### classmethod characters(text)

Returns a list of characters in text.

### classmethod length\_difference(length)

Returns an estimate to a likely change in length relative to an English string of length length.

### classmethod numbertranslate(text)

Converts the numbers in a string according to the rules of the language.

#### classmethod numstart(text)

Determines whether the text starts with a numeric value.

### classmethod sentence\_iter(text, strip=True)

Returns an iterator over the sentences in text.

### classmethod sentences (text, strip=True)

Returns a list of sentences in text.

#### classmethod word iter(text)

Returns an iterator over the words in text.

#### classmethod words(text)

Returns a list of words in text.

#### bn

This module represents the Bengali language.

#### See also:

http://en.wikipedia.org/wiki/Bengali\_language

### class translate.lang.bn.bn

This class represents Bengali.

### classmethod alter\_length(text)

Converts the given string by adding or removing characters as an estimation of translation length (with English assumed as source language).

### classmethod capsstart(text)

Determines whether the text starts with a capital letter.

## classmethod character\_iter(text)

Returns an iterator over the characters in text.

#### classmethod characters(text)

Returns a list of characters in text.

### classmethod length\_difference(length)

Returns an estimate to a likely change in length relative to an English string of length length.

## classmethod numbertranslate(text)

Converts the numbers in a string according to the rules of the language.

#### classmethod numstart(text)

Determines whether the text starts with a numeric value.

#### classmethod punctranslate(text)

Converts the punctuation in a string according to the rules of the language.

# classmethod sentence\_iter(text, strip=True)

Returns an iterator over the sentences in text.

#### classmethod sentences (text, strip=True)

Returns a list of sentences in text.

## classmethod word\_iter(text)

Returns an iterator over the words in text.

### classmethod words(text)

Returns a list of words in text.

#### code or

This module represents the Odia language.

#### See also:

https://en.wikipedia.org/wiki/Odia\_language

### class translate.lang.code\_or.code\_or

This class represents Odia.

### classmethod alter\_length(text)

Converts the given string by adding or removing characters as an estimation of translation length (with English assumed as source language).

#### classmethod capsstart(text)

Determines whether the text starts with a capital letter.

### classmethod character\_iter(text)

Returns an iterator over the characters in text.

### classmethod characters(text)

Returns a list of characters in text.

## classmethod length\_difference(length)

Returns an estimate to a likely change in length relative to an English string of length length.

### classmethod numbertranslate(text)

Converts the numbers in a string according to the rules of the language.

### classmethod numstart(text)

Determines whether the text starts with a numeric value.

#### classmethod punctranslate(text)

Converts the punctuation in a string according to the rules of the language.

## classmethod sentence\_iter(text, strip=True)

Returns an iterator over the sentences in text.

### classmethod sentences(text, strip=True)

Returns a list of sentences in text.

### classmethod word\_iter(text)

Returns an iterator over the words in text.

## classmethod words(text)

Returns a list of words in text.

#### common

This module contains all the common features for languages.

Supported features:

- language code (km, af)
- language name (Khmer, Afrikaans)
- · Plurals
  - Number of plurals (nplurals)
  - Plural equation

• pofilter tests to ignore

### Segmentation:

- · characters
- · words
- · sentences

#### Punctuation:

- · End of sentence
- · Start of sentence
- Middle of sentence
- Ouotes
  - single
  - double
- Valid characters
- · Accelerator characters
- Special characters
- Direction (rtl or ltr)

### TODOs and Ideas for possible features:

- Language-Team information
- · Segmentation
  - phrases

### class translate.lang.common.Common

This class is the common parent class for all language classes.

# CJKpunc = u'\u3002\u3001\uff0c\uff1b\uff01\uff1f\u300c\u300d\u300e\u300f\u3010\u3011'

These punctuation marks are used in certain circumstances with CJK languages.

## classmethod alter\_length(text)

Converts the given string by adding or removing characters as an estimation of translation length (with English assumed as source language).

## classmethod capsstart(text)

Determines whether the text starts with a capital letter.

### classmethod character\_iter(text)

Returns an iterator over the characters in text.

## classmethod characters(text)

Returns a list of characters in text.

#### checker = None

A language specific checker instance (see filters.checks).

This doesn't need to be supplied, but will be used if it exists.

### code = u''

The ISO 639 language code, possibly with a country specifier or other modifier.

Examples:

```
km
pt_BR
sr_YU@Latn
```

## commonpunc = u'.,;:!?-@#\$%^\*\_()[]{}/\\'`"<>'

These punctuation marks are common in English and most languages that use latin script.

### ethiopicpunc = $u'\u1362\u1364\u1363'$

These punctuation marks are used by several Ethiopic languages.

#### fullname = u''

The full (English) name of this language.

Dialect codes should have the form of:

- Khmer
- Portugese (Brazil)
- TODO: sr\_YU@Latn?

### ignoretests = {}

Dictionary of tests to ignore in some or all checkers.

Keys are checker names and values are list of names for the ignored tests in the checker. A special 'all' checker name can be used to tell that the tests must be ignored in all the checkers.

Listed checkers to ignore tests on must be lowercase strings for the checker name, for example "mozilla" for MozillaChecker or "libreoffice" for LibreOfficeChecker.

#### $indicpunc = u' \u0964 \u0965 \u0970'$

These punctuation marks are used by several Indic languages.

#### invertedpunc = u'\xbf\xa1'

Inverted punctuation sometimes used at the beginning of sentences in Spanish, Asturian, Galician, and

### classmethod length\_difference(length)

Returns an estimate to a likely change in length relative to an English string of length length.

## listseperator = u', '

This string is used to separate lists of textual elements. Most languages probably can stick with the default comma, but Arabic and some Asian languages might want to override this.

## miscpunc = $u'\u2026\xb1\xb0\xb9\xb2\xb3\xb7\xa9\xae\xd7\xa3\xa5\u20ac'$

The middle dot  $(\cdot)$  is used by Greek and Georgian.

### mozilla\_pluralequation = u'0'

This of languages that has different plural formula in Mozilla than the standard one in Gettext.

#### nplurals = 0

The number of plural forms of this language.

0 is not a valid value - it must be overridden. Any positive integer is valid (it should probably be between 1 and 6)

#### See also:

```
translate.lang.data
```

## classmethod numbertranslate(text)

Converts the numbers in a string according to the rules of the language.

## numbertuple = ()

A tuple of number transformation rules that can be used by numbertranslate().

#### classmethod numstart(text)

Determines whether the text starts with a numeric value.

#### pluralequation = u'0'

The plural equation for selection of plural forms.

This is used for PO files to fill into the header.

#### See also:

Gettext manual, translate.lang.data

### puncdict = {}

A dictionary of punctuation transformation rules that can be used by punctranslate().

### classmethod punctranslate(text)

Converts the punctuation in a string according to the rules of the language.

## $punctuation = u'.,;: !?-@ \# \$^*_() [] \{ \} / \ '` "<> \ u2018 \ u2019 \ u201b \ u201c \ u201d \ u201e \ u201e$

We include many types of punctuation here, simply since this is only meant to determine if something is punctuation. Hopefully we catch some languages which might not be represented with modules. Most languages won't need to override this.

# $\label{eq:quotes} {\tt quotes} \ = \ {\tt u'} \ {\tt u2018} \ {\tt u201b} \ {\tt u201c} \ {\tt u201d} \ {\tt u201f} \ {\tt u2033} \ {\tt u2034} \ {\tt u2035} \ {\tt u2036} \ {\tt u2036$

These are different quotation marks used by various languages.

### $rtlpunc = u'\u060c\u061f\u061b\xf7'$

These punctuation marks are used by Arabic and Persian, for example.

### classmethod sentence\_iter(text, strip=True)

Returns an iterator over the sentences in text.

## $\tt sentenceend = u'.!?\u2026\u0589\u061f\u0964\u3002\uff01\uff1f\u1362\u06d4'$

These marks can indicate a sentence end. Once again we try to account for many languages. Most languages won't need to override this.

#### classmethod sentences (text, strip=True)

Returns a list of sentences in text.

## specialchars = u''

Characters used by the language that might not be easy to input with common keyboard layouts

#### validaccel = None

Characters that can be used as accelerators (access keys) i.e. Alt+X where X is the accelerator. These can include combining diacritics as long as they are accessible from the users keyboard in a single keystroke, but normally they would be at least precomposed characters. All characters, lower and upper, are included in the list.

## validdoublewords = []

Some languages allow double words in certain cases. This is a dictionary of such words.

### classmethod word\_iter(text)

Returns an iterator over the words in text.

#### classmethod words (text)

Returns a list of words in text.

#### data

This module stores information and functionality that relates to plurals.

translate.lang.data.cldr\_plural\_categories = [u'zero', u'one', u'two', u'few', u'many', u'one', u'few', u'

translate.lang.data.expansion\_factors = {u'af': 0.1, u'ar': -0.09, u'es': 0.21, u'fr': Source to target string length expansion factors.

translate.lang.data.forceunicode(string)

Ensures that the string is in unicode.

Parameters string (Unicode, String) - A text string

**Returns** String converted to Unicode and normalized as needed.

Return type Unicode

translate.lang.data.get\_country\_iso\_name(country\_code)
Return country ISO name.

 $\verb|translate.lang.data.get_language_iso_fullname| (|language_code|)$ 

Return language ISO fullname.

If language code is not a simple ISO 639 code, then we try to split into a two part language code (ISO 639 and ISO 3166).

translate.lang.data.get\_language\_iso\_name(language\_code)
Return language ISO name.

translate.lang.data.gettext\_country(langcode=None)

Returns a gettext function to translate country names into the given language, or the system language if no language is specified.

translate.lang.data.gettext\_domain(langcode, domain, localedir=None)

Returns a gettext function for given iso domain

translate.lang.data.gettext\_lang(langcode=None)

Returns a gettext function to translate language names into the given language, or the system language if no language is specified.

translate.lang.data.languagematch (languagecode, otherlanguagecode) matches a languagecode to another, ignoring regions in the second

translate.lang.data.languages = {u'ach': (u'Acholi', 2, u'n > 1'), u'af': (u'Afrikaans', Dictionary of language data. The language code is the dictionary key (which may contain country codes and modifiers). The value is a tuple: (Full name in English from iso-codes, nplurals, plural equation).

Note that the English names should not be used in user facing places - it should always be passed through the function returned from tr\_lang(), or at least passed through \_fix\_language\_name().

translate.lang.data.normalize(string, normal\_form=u'NFC')

Return a unicode string in its normalized form

#### **Parameters**

- string The string to be normalized
- normal\_form NFC (default), NFD, NFKC, NFKD

**Returns** Normalized string

translate.lang.data.normalized\_unicode(string)

Forces the string to unicode and does normalization.

translate.lang.data.scripts = {u'Beng': [u'bn', u'mni'], u'Deva': [u'anp', u'bho', u'brx Dictionary of scripts data. The dictionary keys are ISO 15924 script codes, and script names where scripts are missing from standard. The value is a list of codes for languages using that script.

This is mainly used to alter the behavior of some checks (the accelerators one for example).

```
translate.lang.data.simplercode(code)
```

This attempts to simplify the given language code by ignoring country codes, for example.

#### See also:

- http://www.rfc-editor.org/rfc/bcp/bcp47.txt
- http://www.rfc-editor.org/rfc/rfc4646.txt
- http://www.rfc-editor.org/rfc/rfc4647.txt
- http://www.w3.org/International/articles/language-tags/

translate.lang.data.simplify\_to\_common(language\_code, languages={u'ach': (u'Acholi', 2, u'n > 1'), u'af': (u'Afrikaans', 2, u'(n != 1)'), u'ak': (u'Akan', 2, u'n > 1'), u'am': (u'Amharic',2, u'n > 1'), u'an': (u'Aragonese', 2, u'(n !=1)'), u'anp': (u'Angika', 2, u'(n != 1)'), u'ar': (u'Arabic', 6, u'n==0 ? 0 : n==1 ? 1 :n==2 ? 2 : n%100>=3 && n%100<=10 ? 3 : n%100 >= 11?4:5'), u'arn': (u'Mapudungun; Mapuche', 2, u'n > 1'), u'as': (u'Assamese', 2, u'(n)!= 1)'), u'ast': (u'Asturian; Bable; Leonese; Asturleonese', 2, u'(n != 1)'), u'ay':  $(u'Aymar \times e1', 1,$ u'0'), u'az': (u'Azerbaijani', 2, u'(n != 1)'), u'be': (u'Belarusian', 3, u'n%10==1 && n%100!=11 ?0: n%10 >= 2 && n%10 <= 4 && (n%100 < 10)n%100>=20) ? 1 : 2'), u'bg': (u'Bulgarian', 2, u'(n != 1)'), u'bn': (u'Bengali', 2, u'(n != 1)'),  $u'bn\_BD'$ : (u'Bengali (Bangladesh)', 2, u'(n!=1)'), *u'bn IN'*: (*u'Bengali* (*India*)', 2, *u'*(*n* != 1)'), *u'bo'*: (u'Tibetan', 1, u'0'), u'br': (u'Breton', 2, u'n > 1'),u'brx': (u'Bodo', 2, u'(n!=1)'), u'bs': (u'Bosnian',3, u'n%10==1 && n%100!=11 ? 0 : n%10>=2&& n%10 <= 4 &&  $(n\%100 < 10 \mid\mid n\%100 >= 20)$ ? 1 : 2'), u'ca': (u'Catalan; Valencian', 2, u'(n != 1)'), u'ca@valencia': (u'Catalan; Valencian (Valencia)', 2, u'(n!=1)'), u'cgg': (u'Chiga', 1, u'0'),u'cs': (u'Czech', 3, u'(n==1)? 0: (n>=2 &&n < = 4) ? 1 : 2'), u'csb': (u'Kashubian', 3, u'n==1 ? 0: n%10>=2 && n%10<=4 && (n%100<10 || n%100 > = 20|? 1: 2'), u'cy': (u'Welsh', 2, u'(n==2) ? 1 : 0'), u'da': (u'Danish', 2, u'(n=1)!= 1)'), u'de': (u'German', 2, u'(n != 1)'), u'doi': (u'Dogri', 2, u'(n != 1)'), u'dz': (u'Dzongkha',1, u'0'), u'el': (u'Greek, Modern (1453-)', 2, u'(n != 1)'), u'en': (u'English', 2, u'(n != 1)'), u'en GB': (u'English (United Kingdom)', 2, u'(n!= 1)'), u'en\_ZA': (u'English (South Africa)', 2, u'(n != 1)'), u'eo': (u'Esperanto', 2, u'(n != 1)'), u'es': (u'Spanish; Castilian', 2, u'(n != 1)'), u'es\_AR': (u'Argentinean Spanish', 2, u'(n != 1)'), u'et':(u'Estonian', 2, u'(n != 1)'), u'eu': (u'Basque',2, u'(n != 1)'), u'fa': (u'Persian', 2, u'n > 1'), u'ff': (u'Fulah', 2, u'(n!=1)'), u'fi': (u'Finnish',2, u'(n != 1)'), u'fil': (u'Filipino; Pilipino', 2, u'(n != 1)')> 1)'), u'fo': (u'Faroese', 2, u'(n != 1)'), u'fr': (u'French', 2, u'(n > 1)'), u'fur': (u'Friulian', 2,u'(n!=1)'), u'fy': (u'Frisian', 2, u'(n!=1)'), u'ga': (u'Irish', 5, u'n==1? 0: n==2? 1: (n>2)&& n < 7)? 2:(n > 6 && n < 11)? 3: 4'), u'gd': (u'Gaelic; Scottish Gaelic', 4, u'( $n==1 \mid \mid n==11$ ) ?  $0: (n==2 \mid \mid n==12)$  ? 1: (n > 2 && n< 20) ? 2 : 3'), u'gl': (u'Galician', 2, u'(n != 1)'), u'gu': (u'Gujarati', 2, u'(n != 1)'), u'gun': (u'Gun', 2, u'(n > 1)'), u'ha': (u'Hausa', 2, u'(n > 1)')!= 1)'), u'he': (u'Hebrew', 2, u'(n != 1)'), u'hi': (u'Hindi', 2, u'(n!=1)'), u'hne': (u'Chhattisgarhi',2, u'(n!=1)'), u'hr': (u'Croatian', 3, u'(n%10==1)')&& n%100!=11 ? 0 : n% Chapter && A.P.I. Reference

> &&  $(n\%100<10 \mid |n\%100>=20)$ ? 1: 2)'), u'ht': (u'Haitian; Haitian Creole', 2, u'(n != 1)'), u'hu': (u'Hungarian', 2, u'(n!=1)'), u'hy': (u'Armenian',

Simplify language code to the most commonly used form for the language, stripping country information for languages that tend not to be localized differently for different countries

```
translate.lang.data.tr_lang(langcode=None)
```

Gives a function that can translate a language name, even in the form "language" (country) ", into the language with iso code language, or the system language if no language is specified.

#### de

This module represents the German language.

#### See also:

http://en.wikipedia.org/wiki/German\_language

## class translate.lang.de.de

This class represents German.

### classmethod alter\_length(text)

Converts the given string by adding or removing characters as an estimation of translation length (with English assumed as source language).

### classmethod capsstart(text)

Determines whether the text starts with a capital letter.

### classmethod character\_iter(text)

Returns an iterator over the characters in text.

#### classmethod characters (text)

Returns a list of characters in text.

### classmethod length\_difference(length)

Returns an estimate to a likely change in length relative to an English string of length length.

### classmethod numbertranslate(text)

Converts the numbers in a string according to the rules of the language.

### classmethod numstart(text)

Determines whether the text starts with a numeric value.

## classmethod punctranslate(text)

Converts the punctuation in a string according to the rules of the language.

### classmethod sentence\_iter(text, strip=True)

Returns an iterator over the sentences in text.

#### classmethod sentences (text, strip=True)

Returns a list of sentences in text.

### classmethod word iter(text)

Returns an iterator over the words in text.

## classmethod words(text)

Returns a list of words in text.

#### el

This module represents the Greek language.

#### See also:

#### http://en.wikipedia.org/wiki/Greek\_language

#### class translate.lang.el.el

This class represents Greek.

### classmethod alter\_length(text)

Converts the given string by adding or removing characters as an estimation of translation length (with English assumed as source language).

#### classmethod capsstart (text)

Determines whether the text starts with a capital letter.

#### classmethod character\_iter(text)

Returns an iterator over the characters in text.

#### classmethod characters(text)

Returns a list of characters in text.

#### classmethod length\_difference(length)

Returns an estimate to a likely change in length relative to an English string of length length.

#### classmethod numbertranslate(text)

Converts the numbers in a string according to the rules of the language.

#### classmethod numstart(text)

Determines whether the text starts with a numeric value.

#### classmethod punctranslate(text)

Converts the punctuation in a string according to the rules of the language.

### classmethod sentence\_iter(text, strip=True)

Returns an iterator over the sentences in text.

### classmethod sentences(text, strip=True)

Returns a list of sentences in text.

#### classmethod word iter(text)

Returns an iterator over the words in text.

### classmethod words (text)

Returns a list of words in text.

#### es

This module represents the Spanish language.

**Note:** As it only has special case code for initial inverted punctuation, it could also be used for Asturian, Galician, or Catalan.

## class translate.lang.es.es

This class represents Spanish.

#### classmethod alter\_length(text)

Converts the given string by adding or removing characters as an estimation of translation length (with English assumed as source language).

## classmethod capsstart(text)

Determines whether the text starts with a capital letter.

#### classmethod character iter(text)

Returns an iterator over the characters in text.

### classmethod characters(text)

Returns a list of characters in text.

### classmethod length\_difference(length)

Returns an estimate to a likely change in length relative to an English string of length length.

#### classmethod numbertranslate(text)

Converts the numbers in a string according to the rules of the language.

### classmethod numstart(text)

Determines whether the text starts with a numeric value.

### classmethod punctranslate(text)

Implement some extra features for inverted punctuation.

## classmethod sentence\_iter(text, strip=True)

Returns an iterator over the sentences in text.

#### classmethod sentences (text, strip=True)

Returns a list of sentences in text.

### classmethod word\_iter(text)

Returns an iterator over the words in text.

### classmethod words(text)

Returns a list of words in text.

### factory

This module provides a factory to instantiate language classes.

```
translate.lang.factory.get_all_languages()
    Return all language classes.
```

```
translate.lang.factory.getlanguage(code)
```

This returns a language class.

Parameters code – The ISO 639 language code

### fa

This module represents the Persian language.

### See also:

http://en.wikipedia.org/wiki/Persian\_language

## class translate.lang.fa.fa

This class represents Persian.

#### classmethod alter\_length(text)

Converts the given string by adding or removing characters as an estimation of translation length (with English assumed as source language).

## classmethod capsstart(text)

Determines whether the text starts with a capital letter.

#### classmethod character iter(text)

Returns an iterator over the characters in text.

### classmethod characters(text)

Returns a list of characters in text.

#### classmethod length\_difference(length)

Returns an estimate to a likely change in length relative to an English string of length length.

#### classmethod numbertranslate(text)

Converts the numbers in a string according to the rules of the language.

### classmethod numstart(text)

Determines whether the text starts with a numeric value.

### classmethod punctranslate(text)

Implement "French" quotation marks.

#### classmethod sentence\_iter(text, strip=True)

Returns an iterator over the sentences in text.

#### classmethod sentences(text, strip=True)

Returns a list of sentences in text.

### classmethod word\_iter(text)

Returns an iterator over the words in text.

#### classmethod words (text)

Returns a list of words in text.

fi

This module represents the Finnish language.

### class translate.lang.fi.fi

This class represents Finnish.

## classmethod alter\_length(text)

Converts the given string by adding or removing characters as an estimation of translation length (with English assumed as source language).

### classmethod capsstart(text)

Determines whether the text starts with a capital letter.

#### classmethod character iter(text)

Returns an iterator over the characters in text.

### classmethod characters(text)

Returns a list of characters in text.

## classmethod length\_difference(length)

Returns an estimate to a likely change in length relative to an English string of length length.

#### classmethod numbertranslate(text)

Converts the numbers in a string according to the rules of the language.

#### classmethod numstart (text)

Determines whether the text starts with a numeric value.

## classmethod punctranslate(text)

Converts the punctuation in a string according to the rules of the language.

#### classmethod sentence iter(text, strip=True)

Returns an iterator over the sentences in text.

### classmethod sentences (text, strip=True)

Returns a list of sentences in text.

### classmethod word\_iter(text)

Returns an iterator over the words in text.

#### classmethod words(text)

Returns a list of words in text.

#### fr

This module represents the French language.

#### See also:

http://en.wikipedia.org/wiki/French\_language

### class translate.lang.fr.fr

This class represents French.

### classmethod alter\_length(text)

Converts the given string by adding or removing characters as an estimation of translation length (with English assumed as source language).

### classmethod capsstart(text)

Determines whether the text starts with a capital letter.

## classmethod character\_iter(text)

Returns an iterator over the characters in text.

#### classmethod characters (text)

Returns a list of characters in text.

#### classmethod length\_difference (length)

Returns an estimate to a likely change in length relative to an English string of length length.

#### classmethod numbertranslate(text)

Converts the numbers in a string according to the rules of the language.

### classmethod numstart(text)

Determines whether the text starts with a numeric value.

### classmethod punctranslate(text)

Implement some extra features for quotation marks.

#### **Known shortcomings:**

- % and \$ are not touched yet for fear of variables
- Double spaces might be introduced

## classmethod sentence\_iter(text, strip=True)

Returns an iterator over the sentences in text.

### classmethod sentences(text, strip=True)

Returns a list of sentences in text.

## classmethod word\_iter(text)

Returns an iterator over the words in text.

#### classmethod words (text)

Returns a list of words in text.

#### gu

This module represents the Gujarati language.

#### See also:

http://en.wikipedia.org/wiki/Gujarati\_language

```
class translate.lang.gu.gu
```

This class represents Gujarati.

#### classmethod alter\_length(text)

Converts the given string by adding or removing characters as an estimation of translation length (with English assumed as source language).

### classmethod capsstart(text)

Determines whether the text starts with a capital letter.

#### classmethod character\_iter(text)

Returns an iterator over the characters in text.

#### classmethod characters (text)

Returns a list of characters in text.

### classmethod length\_difference(length)

Returns an estimate to a likely change in length relative to an English string of length length.

### classmethod numbertranslate(text)

Converts the numbers in a string according to the rules of the language.

## classmethod numstart(text)

Determines whether the text starts with a numeric value.

#### classmethod punctranslate(text)

Converts the punctuation in a string according to the rules of the language.

### classmethod sentence\_iter(text, strip=True)

Returns an iterator over the sentences in text.

### classmethod sentences (text, strip=True)

Returns a list of sentences in text.

#### classmethod word iter(text)

Returns an iterator over the words in text.

### classmethod words(text)

Returns a list of words in text.

### he

This module represents the Hebrew language.

### See also:

http://en.wikipedia.org/wiki/Hebrew\_language

### class translate.lang.he.he

This class represents Hebrew.

#### classmethod alter length (text)

Converts the given string by adding or removing characters as an estimation of translation length (with English assumed as source language).

### classmethod capsstart(text)

Determines whether the text starts with a capital letter.

#### classmethod character iter(text)

Returns an iterator over the characters in text.

### classmethod characters(text)

Returns a list of characters in text.

### classmethod length\_difference(length)

Returns an estimate to a likely change in length relative to an English string of length length.

#### classmethod numbertranslate(text)

Converts the numbers in a string according to the rules of the language.

### classmethod numstart(text)

Determines whether the text starts with a numeric value.

#### classmethod punctranslate(text)

Converts the punctuation in a string according to the rules of the language.

#### classmethod sentence iter(text, strip=True)

Returns an iterator over the sentences in text.

### classmethod sentences (text, strip=True)

Returns a list of sentences in text.

### classmethod word\_iter(text)

Returns an iterator over the words in text.

## classmethod words(text)

Returns a list of words in text.

#### hi

This module represents the Hindi language.

#### See also:

http://en.wikipedia.org/wiki/Hindi\_language

### class translate.lang.hi.hi

This class represents Hindi.

#### classmethod alter\_length (text)

Converts the given string by adding or removing characters as an estimation of translation length (with English assumed as source language).

### classmethod capsstart(text)

Determines whether the text starts with a capital letter.

#### classmethod character iter(text)

Returns an iterator over the characters in text.

## classmethod characters(text)

Returns a list of characters in text.

### classmethod length difference(length)

Returns an estimate to a likely change in length relative to an English string of length length.

#### classmethod numbertranslate(text)

Converts the numbers in a string according to the rules of the language.

#### classmethod numstart(text)

Determines whether the text starts with a numeric value.

#### classmethod punctranslate(text)

Converts the punctuation in a string according to the rules of the language.

### classmethod sentence\_iter(text, strip=True)

Returns an iterator over the sentences in text.

#### classmethod sentences (text, strip=True)

Returns a list of sentences in text.

### classmethod word\_iter(text)

Returns an iterator over the words in text.

#### classmethod words(text)

Returns a list of words in text.

#### hy

This module represents the Armenian language.

#### See also:

http://en.wikipedia.org/wiki/Armenian\_language

### class translate.lang.hy.hy

This class represents Armenian.

### classmethod alter\_length(text)

Converts the given string by adding or removing characters as an estimation of translation length (with English assumed as source language).

#### classmethod capsstart(text)

Determines whether the text starts with a capital letter.

### classmethod character\_iter(text)

Returns an iterator over the characters in text.

### classmethod characters(text)

Returns a list of characters in text.

#### classmethod length\_difference(length)

Returns an estimate to a likely change in length relative to an English string of length length.

## classmethod numbertranslate(text)

Converts the numbers in a string according to the rules of the language.

#### classmethod numstart(text)

Determines whether the text starts with a numeric value.

### classmethod punctranslate(text)

Converts the punctuation in a string according to the rules of the language.

## classmethod sentence\_iter(text, strip=True)

Returns an iterator over the sentences in text.

#### classmethod sentences (text, strip=True)

Returns a list of sentences in text.

#### classmethod word iter(text)

Returns an iterator over the words in text.

### classmethod words(text)

Returns a list of words in text.

### identify

This module contains functions for identifying languages based on language models.

## ja

This module represents the Japanese language.

#### See also:

http://en.wikipedia.org/wiki/Japanese\_language

```
class translate.lang.ja.ja
```

This class represents Japanese.

### classmethod alter\_length(text)

Converts the given string by adding or removing characters as an estimation of translation length (with English assumed as source language).

### classmethod capsstart(text)

Determines whether the text starts with a capital letter.

## classmethod character\_iter(text)

Returns an iterator over the characters in text.

#### classmethod characters(text)

Returns a list of characters in text.

### classmethod length\_difference(length)

Returns an estimate to a likely change in length relative to an English string of length length.

### classmethod numbertranslate(text)

Converts the numbers in a string according to the rules of the language.

#### classmethod numstart(text)

Determines whether the text starts with a numeric value.

#### classmethod punctranslate(text)

Converts the punctuation in a string according to the rules of the language.

### classmethod sentence\_iter(text, strip=True)

Returns an iterator over the sentences in text.

#### classmethod sentences (text, strip=True)

Returns a list of sentences in text.

### classmethod word\_iter(text)

Returns an iterator over the words in text.

### classmethod words(text)

Returns a list of words in text.

#### km

This module represents the Khmer language.

#### See also:

http://en.wikipedia.org/wiki/Khmer\_language

### class translate.lang.km.km

This class represents Khmer.

### classmethod alter\_length(text)

Converts the given string by adding or removing characters as an estimation of translation length (with English assumed as source language).

#### classmethod capsstart(text)

Determines whether the text starts with a capital letter.

### classmethod character\_iter(text)

Returns an iterator over the characters in text.

### classmethod characters(text)

Returns a list of characters in text.

## $khmerpunc = u'\u17d4\u17d5\u17d6\u17d8'$

These marks are only used for Khmer.

### classmethod length\_difference(length)

Returns an estimate to a likely change in length relative to an English string of length length.

#### classmethod numbertranslate(text)

Converts the numbers in a string according to the rules of the language.

#### classmethod numstart(text)

Determines whether the text starts with a numeric value.

### classmethod punctranslate(text)

Converts the punctuation in a string according to the rules of the language.

### classmethod sentence\_iter(text, strip=True)

Returns an iterator over the sentences in text.

#### classmethod sentences (text, strip=True)

Returns a list of sentences in text.

### classmethod word\_iter(text)

Returns an iterator over the words in text.

# ${\tt classmethod\ words}\ (\textit{text})$

Returns a list of words in text.

#### kn

This module represents the Kannada language.

#### See also:

http://en.wikipedia.org/wiki/Kannada\_language

### class translate.lang.kn.kn

This class represents Kannada.

#### classmethod alter length (text)

Converts the given string by adding or removing characters as an estimation of translation length (with English assumed as source language).

### classmethod capsstart(text)

Determines whether the text starts with a capital letter.

#### classmethod character iter(text)

Returns an iterator over the characters in text.

### classmethod characters(text)

Returns a list of characters in text.

### classmethod length\_difference(length)

Returns an estimate to a likely change in length relative to an English string of length length.

#### classmethod numbertranslate(text)

Converts the numbers in a string according to the rules of the language.

### classmethod numstart(text)

Determines whether the text starts with a numeric value.

#### classmethod punctranslate(text)

Converts the punctuation in a string according to the rules of the language.

#### classmethod sentence iter(text, strip=True)

Returns an iterator over the sentences in text.

### classmethod sentences (text, strip=True)

Returns a list of sentences in text.

### classmethod word\_iter(text)

Returns an iterator over the words in text.

## classmethod words(text)

Returns a list of words in text.

## ko

This module represents the Korean language.

#### See also:

http://en.wikipedia.org/wiki/Korean\_language

### class translate.lang.ko.ko

This class represents Korean.

#### classmethod alter\_length(text)

Converts the given string by adding or removing characters as an estimation of translation length (with English assumed as source language).

### classmethod capsstart(text)

Determines whether the text starts with a capital letter.

#### classmethod character iter(text)

Returns an iterator over the characters in text.

## classmethod characters(text)

Returns a list of characters in text.

#### classmethod length difference (length)

Returns an estimate to a likely change in length relative to an English string of length length.

### classmethod numbertranslate(text)

Converts the numbers in a string according to the rules of the language.

#### classmethod numstart(text)

Determines whether the text starts with a numeric value.

#### classmethod punctranslate(text)

Converts the punctuation in a string according to the rules of the language.

### classmethod sentence\_iter(text, strip=True)

Returns an iterator over the sentences in text.

#### classmethod sentences (text, strip=True)

Returns a list of sentences in text.

### classmethod word\_iter(text)

Returns an iterator over the words in text.

#### classmethod words(text)

Returns a list of words in text.

#### ml

This module represents the Malayalam language.

#### See also:

http://en.wikipedia.org/wiki/Malayalam\_language

### class translate.lang.ml.ml

This class represents Malayalam.

### classmethod alter\_length(text)

Converts the given string by adding or removing characters as an estimation of translation length (with English assumed as source language).

#### classmethod capsstart(text)

Determines whether the text starts with a capital letter.

### classmethod character\_iter(text)

Returns an iterator over the characters in text.

### classmethod characters(text)

Returns a list of characters in text.

#### classmethod length\_difference(length)

Returns an estimate to a likely change in length relative to an English string of length length.

### classmethod numbertranslate(text)

Converts the numbers in a string according to the rules of the language.

#### classmethod numstart(text)

Determines whether the text starts with a numeric value.

### classmethod punctranslate(text)

Converts the punctuation in a string according to the rules of the language.

## classmethod sentence\_iter(text, strip=True)

Returns an iterator over the sentences in text.

#### classmethod sentences (text, strip=True)

Returns a list of sentences in text.

#### classmethod word iter(text)

Returns an iterator over the words in text.

## classmethod words(text)

Returns a list of words in text.

#### mr

This module represents the Marathi language.

### See also:

http://en.wikipedia.org/wiki/Marathi\_language

## class translate.lang.mr.mr

This class represents Marathi.

### classmethod alter\_length(text)

Converts the given string by adding or removing characters as an estimation of translation length (with English assumed as source language).

### classmethod capsstart(text)

Determines whether the text starts with a capital letter.

### classmethod character\_iter(text)

Returns an iterator over the characters in text.

## classmethod characters(text)

Returns a list of characters in text.

## ${\tt classmethod\ length\_difference}\ (\mathit{length})$

Returns an estimate to a likely change in length relative to an English string of length length.

#### classmethod numbertranslate(text)

Converts the numbers in a string according to the rules of the language.

### classmethod numstart(text)

Determines whether the text starts with a numeric value.

### classmethod punctranslate(text)

Converts the punctuation in a string according to the rules of the language.

### classmethod sentence\_iter(text, strip=True)

Returns an iterator over the sentences in text.

#### classmethod sentences (text, strip=True)

Returns a list of sentences in text.

## classmethod word\_iter(text)

Returns an iterator over the words in text.

#### classmethod words(text)

Returns a list of words in text.

#### ne

This module represents the Nepali language.

#### See also:

http://en.wikipedia.org/wiki/Nepali\_language

### class translate.lang.ne.ne

This class represents Nepali.

### classmethod alter\_length(text)

Converts the given string by adding or removing characters as an estimation of translation length (with English assumed as source language).

## classmethod capsstart(text)

Determines whether the text starts with a capital letter.

#### classmethod character iter(text)

Returns an iterator over the characters in text.

#### classmethod characters(text)

Returns a list of characters in text.

### classmethod length\_difference(length)

Returns an estimate to a likely change in length relative to an English string of length length.

#### classmethod numbertranslate(text)

Converts the numbers in a string according to the rules of the language.

#### classmethod numstart(text)

Determines whether the text starts with a numeric value.

### classmethod punctranslate(text)

Converts the punctuation in a string according to the rules of the language.

## classmethod sentence\_iter(text, strip=True)

Returns an iterator over the sentences in text.

## classmethod sentences(text, strip=True)

Returns a list of sentences in text.

### classmethod word\_iter(text)

Returns an iterator over the words in text.

### classmethod words(text)

Returns a list of words in text.

### ngram

Ngram models for language guessing.

**Note:** Original code from http://thomas.mangin.me.uk/data/source/ngram.py

#### pa

This module represents the Punjabi language.

## See also:

http://en.wikipedia.org/wiki/Punjabi language

#### class translate.lang.pa.pa

This class represents Punjabi.

### classmethod alter\_length(text)

Converts the given string by adding or removing characters as an estimation of translation length (with English assumed as source language).

#### classmethod capsstart(text)

Determines whether the text starts with a capital letter.

### classmethod character\_iter(text)

Returns an iterator over the characters in text.

### classmethod characters(text)

Returns a list of characters in text.

#### classmethod length\_difference(length)

Returns an estimate to a likely change in length relative to an English string of length length.

### classmethod numbertranslate(text)

Converts the numbers in a string according to the rules of the language.

#### classmethod numstart (text)

Determines whether the text starts with a numeric value.

#### classmethod punctranslate(text)

Converts the punctuation in a string according to the rules of the language.

### classmethod sentence\_iter(text, strip=True)

Returns an iterator over the sentences in text.

### classmethod sentences (text, strip=True)

Returns a list of sentences in text.

## classmethod word\_iter(text)

Returns an iterator over the words in text.

## classmethod words(text)

Returns a list of words in text.

### poedit

Functions to manage Poedit's language features.

**Note:** The ISO 639 maps are from Poedit's isocode.cpp (v1.4.2) to ensure that we match currently released versions of Poedit.

```
or Foedit.
```

translate.lang.poedit.dialects = {u'Chinese': {u'CHINA': u'zh\_CN', u'None': u'zh\_CN', u'}

translate.lang.poedit.isocode(language, country=None)

Returns a language code for the given Poedit language name.

Poedit uses language and country names in the PO header entries:

Language dialects based on ISO 3166 country names, 'None' is the default fallback

- X-Poedit-Language
- X-Poedit-Country

This function converts the supplied language name into the required ISO 639 code. If needed, in the case of dialects, the country name is used to create an xx\_YY style dialect code.

#### **Parameters**

- language (String) Language name
- country (String) Country name

Returns ISO 639 language code

### Return type String

```
translate.lang.poedit.lang_codes = {u'aa': u'Afar', u'ab': u'Abkhazian', u'ae': u'Avesta ISO369 codes and names as used by Poedit. Mostly these are identical to ISO 639, but there are some differences.
```

```
translate.lang.poedit.lang_names = {u'(Afan) Oromo': u'om', u'Abkhazian': u'ab', u'Afar' Reversed lang_codes
```

si

This module represents the Sinhala language.

#### See also:

http://en.wikipedia.org/wiki/Sinhala\_language

```
class translate.lang.si.si
```

This class represents Sinhala.

### classmethod alter\_length(text)

Converts the given string by adding or removing characters as an estimation of translation length (with English assumed as source language).

### classmethod capsstart(text)

Determines whether the text starts with a capital letter.

### classmethod character\_iter(text)

Returns an iterator over the characters in text.

## classmethod characters(text)

Returns a list of characters in text.

## ${\tt classmethod\ length\_difference}\ (\mathit{length})$

Returns an estimate to a likely change in length relative to an English string of length length.

#### classmethod numbertranslate(text)

Converts the numbers in a string according to the rules of the language.

### classmethod numstart(text)

Determines whether the text starts with a numeric value.

## ${\tt classmethod\ punctranslate}\ (\textit{text})$

Converts the punctuation in a string according to the rules of the language.

#### classmethod sentence iter(text, strip=True)

Returns an iterator over the sentences in text.

### classmethod sentences(text, strip=True)

Returns a list of sentences in text.

### classmethod word\_iter(text)

Returns an iterator over the words in text.

### classmethod words(text)

Returns a list of words in text.

#### st

This module represents the Southern Sotho language.

## class translate.lang.st.st

This class represents Southern Sotho.

### classmethod alter\_length(text)

Converts the given string by adding or removing characters as an estimation of translation length (with English assumed as source language).

### classmethod capsstart(text)

Determines whether the text starts with a capital letter.

#### classmethod character\_iter(text)

Returns an iterator over the characters in text.

#### classmethod characters(text)

Returns a list of characters in text.

### classmethod length\_difference(length)

Returns an estimate to a likely change in length relative to an English string of length length.

## classmethod numbertranslate(text)

Converts the numbers in a string according to the rules of the language.

## classmethod numstart(text)

Determines whether the text starts with a numeric value.

### classmethod punctranslate(text)

Converts the punctuation in a string according to the rules of the language.

### classmethod sentence\_iter(text, strip=True)

Returns an iterator over the sentences in text.

### classmethod sentences(text, strip=True)

Returns a list of sentences in text.

### classmethod word iter(text)

Returns an iterator over the words in text.

## classmethod words(text)

Returns a list of words in text.

#### sv

This module represents the the Swedish language.

### See also:

http://en.wikipedia.org/wiki/Swedish\_language

### class translate.lang.sv.sv

This class represents Swedish.

#### classmethod alter length (text)

Converts the given string by adding or removing characters as an estimation of translation length (with English assumed as source language).

### classmethod capsstart(text)

Determines whether the text starts with a capital letter.

#### classmethod character iter(text)

Returns an iterator over the characters in text.

### classmethod characters(text)

Returns a list of characters in text.

### classmethod length\_difference(length)

Returns an estimate to a likely change in length relative to an English string of length length.

#### classmethod numbertranslate(text)

Converts the numbers in a string according to the rules of the language.

### classmethod numstart(text)

Determines whether the text starts with a numeric value.

#### classmethod punctranslate(text)

Converts the punctuation in a string according to the rules of the language.

#### classmethod sentence iter(text, strip=True)

Returns an iterator over the sentences in text.

### classmethod sentences (text, strip=True)

Returns a list of sentences in text.

### classmethod word\_iter(text)

Returns an iterator over the words in text.

## classmethod words(text)

Returns a list of words in text.

### ta

This module represents the Tamil language.

#### See also:

http://en.wikipedia.org/wiki/Tamil\_language

### class translate.lang.ta.ta

This class represents Tamil.

#### classmethod alter\_length(text)

Converts the given string by adding or removing characters as an estimation of translation length (with English assumed as source language).

### classmethod capsstart (text)

Determines whether the text starts with a capital letter.

#### classmethod character iter(text)

Returns an iterator over the characters in text.

## classmethod characters(text)

Returns a list of characters in text.

#### classmethod length\_difference(length)

Returns an estimate to a likely change in length relative to an English string of length length.

### classmethod numbertranslate(text)

Converts the numbers in a string according to the rules of the language.

### classmethod numstart(text)

Determines whether the text starts with a numeric value.

#### classmethod punctranslate (text)

Converts the punctuation in a string according to the rules of the language.

### classmethod sentence\_iter(text, strip=True)

Returns an iterator over the sentences in text.

### classmethod sentences (text, strip=True)

Returns a list of sentences in text.

#### classmethod word\_iter(text)

Returns an iterator over the words in text.

#### classmethod words(text)

Returns a list of words in text.

#### team

Module to guess the language ISO code based on the 'Language-Team' entry in the header of a Gettext PO file.

```
translate.lang.team.LANG_TEAM_CONTACT_SNIPPETS = {u'af': (u'i18n@af.org.za', u'Petri Joost Language codes with snippets of contact information that can be used to uniquely identify the language
```

```
translate.lang.team.guess_language(team_string)
```

Gueses the language of a PO file based on the Language-Team entry

### te

This module represents the Telugu language.

### See also:

http://en.wikipedia.org/wiki/Telugu\_language

## class translate.lang.te.te

This class represents Telugu.

### classmethod alter\_length(text)

Converts the given string by adding or removing characters as an estimation of translation length (with English assumed as source language).

## classmethod capsstart(text)

Determines whether the text starts with a capital letter.

#### classmethod character\_iter(text)

Returns an iterator over the characters in text.

## classmethod characters(text)

Returns a list of characters in text.

### classmethod length difference(length)

Returns an estimate to a likely change in length relative to an English string of length length.

#### classmethod numbertranslate(text)

Converts the numbers in a string according to the rules of the language.

### classmethod numstart(text)

Determines whether the text starts with a numeric value.

#### classmethod punctranslate(text)

Converts the punctuation in a string according to the rules of the language.

#### classmethod sentence iter(text, strip=True)

Returns an iterator over the sentences in text.

#### classmethod sentences (text, strip=True)

Returns a list of sentences in text.

### classmethod word\_iter(text)

Returns an iterator over the words in text.

### classmethod words(text)

Returns a list of words in text.

#### th

This module represents the Thai language.

#### See also:

http://en.wikipedia.org/wiki/Thai\_language

### class translate.lang.th.th

This class represents Thai.

### classmethod alter\_length(text)

Converts the given string by adding or removing characters as an estimation of translation length (with English assumed as source language).

### classmethod capsstart(text)

Determines whether the text starts with a capital letter.

### classmethod character\_iter(text)

Returns an iterator over the characters in text.

#### classmethod characters (text)

Returns a list of characters in text.

### classmethod length\_difference(length)

Returns an estimate to a likely change in length relative to an English string of length length.

#### classmethod numbertranslate(text)

Converts the numbers in a string according to the rules of the language.

### classmethod numstart(text)

Determines whether the text starts with a numeric value.

#### classmethod punctranslate(text)

Converts the punctuation in a string according to the rules of the language.

### classmethod sentence\_iter(text, strip=True)

Returns an iterator over the sentences in text.

### classmethod sentences (text, strip=True)

Returns a list of sentences in text.

#### classmethod word\_iter(text)

Returns an iterator over the words in text.

#### classmethod words(text)

Returns a list of words in text.

#### ug

This module represents the Uyghur language.

#### See also:

http://en.wikipedia.org/wiki/Uyghur\_language

#### class translate.lang.ug.ug

This class represents Uyghur.

## classmethod alter\_length(text)

Converts the given string by adding or removing characters as an estimation of translation length (with English assumed as source language).

#### classmethod capsstart(text)

Determines whether the text starts with a capital letter.

## classmethod character\_iter(text)

Returns an iterator over the characters in text.

## classmethod characters(text)

Returns a list of characters in text.

### classmethod length\_difference (length)

Returns an estimate to a likely change in length relative to an English string of length length.

## classmethod numbertranslate(text)

Converts the numbers in a string according to the rules of the language.

### classmethod numstart(text)

Determines whether the text starts with a numeric value.

### classmethod punctranslate(text)

Converts the punctuation in a string according to the rules of the language.

## classmethod sentence\_iter(text, strip=True)

Returns an iterator over the sentences in text.

### classmethod sentences (text, strip=True)

Returns a list of sentences in text.

#### classmethod word\_iter(text)

Returns an iterator over the words in text.

#### classmethod words(text)

Returns a list of words in text.

### ur

This module represents the Urdu language.

### See also:

http://en.wikipedia.org/wiki/Urdu language

#### class translate.lang.ur.ur

This class represents Urdu.

### classmethod alter\_length(text)

Converts the given string by adding or removing characters as an estimation of translation length (with English assumed as source language).

#### classmethod capsstart(text)

Determines whether the text starts with a capital letter.

### classmethod character\_iter(text)

Returns an iterator over the characters in text.

### classmethod characters(text)

Returns a list of characters in text.

#### classmethod length\_difference(length)

Returns an estimate to a likely change in length relative to an English string of length length.

#### classmethod numbertranslate(text)

Converts the numbers in a string according to the rules of the language.

#### classmethod numstart (text)

Determines whether the text starts with a numeric value.

#### classmethod punctranslate(text)

Converts the punctuation in a string according to the rules of the language.

### classmethod sentence\_iter(text, strip=True)

Returns an iterator over the sentences in text.

#### classmethod sentences (text, strip=True)

Returns a list of sentences in text.

## classmethod word\_iter(text)

Returns an iterator over the words in text.

## classmethod words(text)

Returns a list of words in text.

#### νi

This module represents the Vietnamese language.

#### See also:

http://en.wikipedia.org/wiki/Vietnamese language

#### class translate.lang.vi.vi

This class represents Vietnamese.

## classmethod alter\_length(text)

Converts the given string by adding or removing characters as an estimation of translation length (with English assumed as source language).

#### classmethod capsstart(text)

Determines whether the text starts with a capital letter.

## classmethod character\_iter(text)

Returns an iterator over the characters in text.

#### classmethod characters (text)

Returns a list of characters in text.

### classmethod length\_difference(length)

Returns an estimate to a likely change in length relative to an English string of length length.

#### classmethod numbertranslate(text)

Converts the numbers in a string according to the rules of the language.

#### classmethod numstart(text)

Determines whether the text starts with a numeric value.

## classmethod punctranslate(text)

Implement some extra features for quotation marks.

### **Known shortcomings:**

- % and \$ are not touched yet for fear of variables
- · Double spaces might be introduced

#### classmethod sentence iter(text, strip=True)

Returns an iterator over the sentences in text.

## classmethod sentences (text, strip=True)

Returns a list of sentences in text.

### classmethod word\_iter(text)

Returns an iterator over the words in text.

#### classmethod words(text)

Returns a list of words in text.

#### zh

This module represents the Chinese language (Both tradisional and simplified).

#### See also:

http://en.wikipedia.org/wiki/Chinese\_language

## class translate.lang.zh.zh

This class represents Chinese.

### classmethod alter\_length(text)

Converts the given string by adding or removing characters as an estimation of translation length (with English assumed as source language).

#### classmethod capsstart(text)

Determines whether the text starts with a capital letter.

### classmethod character\_iter(text)

Returns an iterator over the characters in text.

#### classmethod characters(text)

Returns a list of characters in text.

### classmethod numbertranslate(text)

Converts the numbers in a string according to the rules of the language.

### classmethod numstart(text)

Determines whether the text starts with a numeric value.

Returns a list of words in text.

```
classmethod punctranslate (text)
    Converts the punctuation in a string according to the rules of the language.
classmethod sentence_iter (text, strip=True)
    Returns an iterator over the sentences in text.
classmethod sentences (text, strip=True)
    Returns a list of sentences in text.
classmethod word_iter (text)
    Returns an iterator over the words in text.
classmethod words (text)
```

#### misc

Miscellaneous modules for translate - including modules for backward compatibility with pre-2.3 versions of Python

### dictutils

```
Implements a case-insensitive (on keys) dictionary and order-sensitive dictionary
class translate.misc.dictutils.cidict(fromdict=None)
      clear() \rightarrow None. Remove all items from D.
      copy() \rightarrow a \text{ shallow copy of } D
      fromkeys (S|, v|) \rightarrow \text{New dict with keys from S and values equal to v.}
             v defaults to None.
      get (k[,d]) \rightarrow D[k] if k in D, else d. d defaults to None.
      has_key (k) \rightarrow True if D has a key k, else False
      items () \rightarrow list of D's (key, value) pairs, as 2-tuples
      iteritems () \rightarrow an iterator over the (key, value) items of D
      iterkeys () \rightarrow an iterator over the keys of D
      \textbf{itervalues} \; (\,) \; \rightarrow \text{an iterator over the values of } D
      keys () \rightarrow list of D's keys
      pop(k|, d|) \rightarrow v, remove specified key and return the corresponding value.
            If key is not found, d is returned if given, otherwise KeyError is raised
      popitem () \rightarrow (k, v), remove and return some (key, value) pair as a
             2-tuple; but raise KeyError if D is empty.
      setdefault (k[,d]) \rightarrow D.get(k,d), also set D[k]=d if k not in D
      update (E) \rightarrow None.
             Update D from E: for k in E.keys(): D[k] = E[k]
      values () \rightarrow list of D's values
      viewitems () \rightarrow a set-like object providing a view on D's items
```

**viewkeys** ()  $\rightarrow$  a set-like object providing a view on D's keys

**viewvalues** ()  $\rightarrow$  an object providing a view on D's values

### file\_discovery

translate.misc.file\_discovery.get\_abs\_data\_filename (path\_parts, basedirs=None)

Get the absolute path to the given file- or directory name in the current running application's data directory.

**Parameters** path\_parts (list) - The path parts that can be joined by os.path.join().

#### Iru

Caching dictionary like object that discards the least recently used objects when number of cached items exceeds maxsize.

cullsize is the fraction of items that will be discarded when maxsize is reached.

#### cull()

free memory by deleting old items from cache

#### itervaluerefs()

Return an iterator that yields the weak references to the values.

The references are not guaranteed to be 'live' at the time they are used, so the result of calling the references needs to be checked before being used. This can be used to avoid creating references that will cause the garbage collector to keep the values around longer than needed.

#### valuerefs()

Return a list of weak references to the values.

The references are not guaranteed to be 'live' at the time they are used, so the result of calling the references needs to be checked before being used. This can be used to avoid creating references that will cause the garbage collector to keep the values around longer than needed.

## multistring

Supports a hybrid Unicode string that can also have a list of alternate strings in the strings attribute

```
class translate.misc.multistring.multistring(*args, **kwargs)
```

```
capitalize() \rightarrow unicode
```

Return a capitalized version of S, i.e. make the first character have upper case and the rest lower case.

```
center (width |, fillchar |) \rightarrow unicode
```

Return S centered in a Unicode string of length width. Padding is done using the specified fill character (default is a space)

```
count (sub[, start[, end]]) \rightarrow int
```

Return the number of non-overlapping occurrences of substring sub in Unicode string S[start:end]. Optional arguments start and end are interpreted as in slice notation.

```
decode ([encoding[, errors]]) \rightarrow string or unicode
```

Decodes S using the codec registered for encoding. encoding defaults to the default encoding. errors may be given to set a different error handling scheme. Default is 'strict' meaning that encoding errors raise a

UnicodeDecodeError. Other possible values are 'ignore' and 'replace' as well as any other name registered with codecs.register error that is able to handle UnicodeDecodeErrors.

# **encode** ([encoding[, errors]]) $\rightarrow$ string or unicode

Encodes S using the codec registered for encoding, encoding defaults to the default encoding, errors may be given to set a different error handling scheme. Default is 'strict' meaning that encoding errors raise a UnicodeEncodeError. Other possible values are 'ignore', 'replace' and 'xmlcharrefreplace' as well as any other name registered with codecs, register error that can handle UnicodeEncodeErrors.

# endswith $(suffix[, start[, end]]) \rightarrow bool$

Return True if S ends with the specified suffix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. suffix can also be a tuple of strings to try.

# expandtabs ([tabsize]) $\rightarrow$ unicode

Return a copy of S where all tab characters are expanded using spaces. If tabsize is not given, a tab size of 8 characters is assumed.

# $find(sub[, start[, end]]) \rightarrow int$

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

### **format** (\*args, \*\*kwargs) → unicode

Return a formatted version of S, using substitutions from args and kwargs. The substitutions are identified by braces ('{ and '}').

# $index(sub[, start[, end]]) \rightarrow int$

Like S.find() but raise ValueError when the substring is not found.

### $isalnum() \rightarrow bool$

Return True if all characters in S are alphanumeric and there is at least one character in S, False otherwise.

### $isalpha() \rightarrow bool$

Return True if all characters in S are alphabetic and there is at least one character in S, False otherwise.

#### $isdecimal() \rightarrow bool$

Return True if there are only decimal characters in S, False otherwise.

### $isdigit() \rightarrow bool$

Return True if all characters in S are digits and there is at least one character in S, False otherwise.

#### $islower() \rightarrow bool$

Return True if all cased characters in S are lowercase and there is at least one cased character in S, False otherwise.

#### $isnumeric() \rightarrow bool$

Return True if there are only numeric characters in S, False otherwise.

## $isspace() \rightarrow bool$

Return True if all characters in S are whitespace and there is at least one character in S, False otherwise.

### $istitle() \rightarrow bool$

Return True if S is a titlecased string and there is at least one character in S, i.e. upper- and titlecase characters may only follow uncased characters and lowercase characters only cased ones. Return False otherwise.

### $isupper() \rightarrow bool$

Return True if all cased characters in S are uppercase and there is at least one cased character in S, False otherwise.

 $join(iterable) \rightarrow unicode$ 

Return a string which is the concatenation of the strings in the iterable. The separator between elements is S.

 $ljust(width[,fillchar]) \rightarrow int$ 

Return S left-justified in a Unicode string of length width. Padding is done using the specified fill character (default is a space).

 $lower() \rightarrow unicode$ 

Return a copy of the string S converted to lowercase.

**lstrip** ([chars])  $\rightarrow$  unicode

Return a copy of the string S with leading whitespace removed. If chars is given and not None, remove characters in chars instead. If chars is a str, it will be converted to unicode before stripping

partition (sep) -> (head, sep, tail)

Search for the separator sep in S, and return the part before it, the separator itself, and the part after it. If the separator is not found, return S and two empty strings.

 $replace(old, new[, count]) \rightarrow unicode$ 

Return a copy of S with all occurrences of substring old replaced by new. If the optional argument count is given, only the first count occurrences are replaced.

**rfind**  $(sub[, start[, end]]) \rightarrow int$ 

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

 $\mathbf{rindex}\,(\mathit{sub}\big[,\mathit{start}\big[,\mathit{end}\,\big]\big])\,\rightarrow\mathrm{int}$ 

Like S.rfind() but raise ValueError when the substring is not found.

 $rjust(width[, fillchar]) \rightarrow unicode$ 

Return S right-justified in a Unicode string of length width. Padding is done using the specified fill character (default is a space).

rpartition (sep) -> (head, sep, tail)

Search for the separator sep in S, starting at the end of S, and return the part before it, the separator itself, and the part after it. If the separator is not found, return two empty strings and S.

 $rsplit([sep[, maxsplit]]) \rightarrow list of strings$ 

Return a list of the words in S, using sep as the delimiter string, starting at the end of the string and working to the front. If maxsplit is given, at most maxsplit splits are done. If sep is not specified, any whitespace string is a separator.

 $\mathbf{rstrip}\,(\big[\mathit{chars}\,\big])\,\to \mathsf{unicode}$ 

Return a copy of the string S with trailing whitespace removed. If chars is given and not None, remove characters in chars instead. If chars is a str, it will be converted to unicode before stripping

 $split([sep[, maxsplit]]) \rightarrow list of strings$ 

Return a list of the words in S, using sep as the delimiter string. If maxsplit is given, at most maxsplit splits are done. If sep is not specified or is None, any whitespace string is a separator and empty strings are removed from the result.

**splitlines** (keepends=False)  $\rightarrow$  list of strings

Return a list of the lines in S, breaking at line boundaries. Line breaks are not included in the resulting list unless keepends is given and true.

 $startswith(prefix[,start[,end]]) \rightarrow bool$ 

Return True if S starts with the specified prefix, False otherwise. With optional start, test S beginning at

that position. With optional end, stop comparing S at that position. prefix can also be a tuple of strings to try.

```
strip([chars]) \rightarrow unicode
```

Return a copy of the string S with leading and trailing whitespace removed. If chars is given and not None, remove characters in chars instead. If chars is a str, it will be converted to unicode before stripping

```
swapcase() \rightarrow unicode
```

Return a copy of S with uppercase characters converted to lowercase and vice versa.

```
title() \rightarrow unicode
```

Return a titlecased version of S, i.e. words start with title case characters, all remaining cased characters have lower case.

```
translate(table) \rightarrow unicode
```

Return a copy of the string S, where all characters have been mapped through the given translation table, which must be a mapping of Unicode ordinals to Unicode ordinals, Unicode strings or None. Unmapped characters are left untouched. Characters mapped to None are deleted.

```
\mathbf{upper}() \rightarrow \mathbf{unicode}
```

Return a copy of S converted to uppercase.

```
zfill (width) \rightarrow unicode
```

Pad a numeric string S with zeros on the left, to fill a field of the specified width. The string S is never truncated.

### optrecurse

A specialized Option Parser for recursing through directories.

```
add_option(Option)
    add_option(opt_str, ..., kwarg=val, ...)
check_values(values: Values, args: [string])
    -> (values: Values, args: [string])
```

Check that the supplied option values and leftover arguments are valid. Returns the option values and leftover arguments (possibly adjusted, possibly completely new – whatever you like). Default implementation just returns the passed-in values; subclasses may override as desired.

```
checkoutputsubdir(options, subdir)
```

Checks to see if subdir under options.output needs to be created, creates if neccessary.

```
define_option(option)
```

Defines the given option, replacing an existing one of the same short name if neccessary...

```
destroy()
```

Declare that you are done with this OptionParser. This cleans up reference cycles so the OptionParser (and all objects referenced by it) can be garbage-collected promptly. After calling destroy(), the OptionParser is unusable.

description=None)

### disable\_interspersed\_args()

Set parsing to stop on the first non-option. Use this if you have a command processor which runs another command that has options of its own and you want to make sure these options don't get confused.

#### enable\_interspersed\_args()

Set parsing to not stop on the first non-option, allowing interspersing switches with command arguments. This is the default behavior. See also disable\_interspersed\_args() and the class documentation description of the attribute allow\_interspersed\_args.

### error (msg : string)

Print a usage message incorporating 'msg' to stderr and exit. If you override this in a subclass, it should not return – it should either exit or raise an exception.

### finalizetempoutputfile (options, outputfile, fulloutputpath)

Write the temp outputfile to its final destination.

### format\_manpage()

returns a formatted manpage

### getformathelp (formats)

Make a nice help string for describing formats...

### getfullinputpath (options, inputpath)

Gets the absolute path to an input file.

## getfulloutputpath (options, outputpath)

Gets the absolute path to an output file.

### getfulltemplatepath (options, templatepath)

Gets the absolute path to a template file.

### getoutputname (options, inputname, outputformat)

Gets an output filename based on the input filename.

## getoutputoptions (options, inputpath, templatepath)

Works out which output format and processor method to use...

## ${\tt getpassthroughoptions}\ (\textit{options})$

Get the options required to pass to the filtermethod...

#### gettemplatename (options, inputname)

Gets an output filename based on the input filename.

### getusageman (option)

returns the usage string for the given option

### getusagestring(option)

returns the usage string for the given option

#### isexcluded(options, inputpath)

Checks if this path has been excluded.

## isrecursive (fileoption, filepurpose='input')

Checks if fileoption is a recursive file.

## isvalidinputname (inputname)

Checks if this is a valid input filename.

### mkdir (parent, subdir)

Makes a subdirectory (recursively if neccessary).

## ${\tt openinputfile} \ (options, full input path)$

Opens the input file.

#### openoutputfile (options, fulloutputpath)

Opens the output file.

### opentemplatefile (options, fulltemplatepath)

Opens the template file (if required).

### opentempoutputfile (options, fulloutputpath)

Opens a temporary output file.

#### parse args (args=None, values=None)

Parses the command line options, handling implicit input/output args.

### print\_help (file : file = stdout)

Print an extended help message, listing all options and any help text provided with them, to 'file' (default stdout).

#### print\_manpage (file=None)

outputs a manpage for the program using the help information

### print\_usage (file : file = stdout)

Print the usage message for the current program (self.usage) to 'file' (default stdout). Any occurrence of the string "%prog" in self.usage is replaced with the name of the current program (basename of sys.argv[0]). Does nothing if self.usage is empty or not defined.

### print\_version (file : file = stdout)

Print the version message for this program (self.version) to 'file' (default stdout). As with print\_usage(), any occurrence of "%prog" in self.version is replaced by the current program's name. Does nothing if self.version is empty or undefined.

### processfile (fileprocessor, options, fullinputpath, fulloutputpath, fulltemplatepath)

Process an individual file.

### recurseinputfilelist (options)

Use a list of files, and find a common base directory for them.

### recurseinputfiles (options)

Recurse through directories and return files to be processed.

#### recursiveprocess (options)

Recurse through directories and process files.

## run()

Parses the arguments, and runs recursive process with the resulting options...

### set\_usage (usage=None)

sets the usage string - if usage not given, uses getusagestring for each option

#### seterrorleveloptions()

Sets the errorlevel options.

## setformats (formats, usetemplates)

Sets the format options using the given format dictionary.

## **Parameters** formats (Dictionary) – The dictionary keys should be:

- Single strings (or 1-tuples) containing an input format (if not *usetemplates*)
- Tuples containing an input format and template format (if *usetemplates*)
- Formats can be *None* to indicate what to do with standard input

The dictionary *values* should be tuples of outputformat (string) and processor method.

```
setmanpageoption()
     creates a manpage option that allows the optionparser to generate a manpage
setprogressoptions()
     Sets the progress options.
splitext (pathname)
     Splits pathname into name and ext, and removes the extsep.
         Parameters pathname (string) – A file path
         Returns root, ext
         Return type tuple
splitinputext (inputpath)
     Splits an inputpath into name and extension.
splittemplateext (templatepath)
     Splits a templatepath into name and extension.
templateexists (options, templatepath)
     Returns whether the given template exists...
warning (msg, options=None, exc_info=None)
     Print a warning message incorporating 'msg' to stderr and exit.
```

### ourdom

module that provides modified DOM functionality for our needs

Note that users of ourdom should ensure that no code might still use classes directly from minidom, like minidom. Element, minidom. Document or methods such as minidom. parseString, since the functionality provided here will not be in those objects.

```
class translate.misc.ourdom.DOMImplementation(*args, **kwargs)
class translate.misc.ourdom.Document(*args, **kwargs)
class translate.misc.ourdom.Element(*args, **kwargs)
class translate.misc.ourdom.ExpatBuilderNS(*args, **kwargs)
     reset()
         Free all data structures used during DOM construction.
translate.misc.ourdom.qetElementsByTaqName helper(parent, name, dummy=None)
     A reimplementation of getElementsByTagName as an iterator.
     Note that this is not compatible with getElementsByTagName that returns a list, therefore, the class below
     exposes this through yieldElementsByTagName
translate.misc.ourdom.getnodetext(node)
     returns the node's text by iterating through the child nodes
translate.misc.ourdom.parse(file, parser=None, bufsize=None)
     Parse a file into a DOM by filename or file object.
translate.misc.ourdom.parseString(string, parser=None)
     Parse a file into a DOM from a string.
```

```
translate.misc.ourdom.searchElementsByTagName_helper(parent, name, onlysearch)
     limits the search to within tags occuring in onlysearch
translate.misc.ourdom.writexml_helper(self, writer, indent=", addindent=", newl=")
     A replacement for writexml that formats it like typical XML files. Nodes are intendented but text nodes, where
     whitespace can be significant, are not indented.
progressbar
Progress bar utilities for reporting feedback on the progress of an application.
class translate.misc.progressbar.DotsProgressBar
     An ultra-simple progress indicator that just writes a dot for each action
     show (verbosemessage)
          show a dot for progress :-)
class translate.misc.progressbar.HashProgressBar(*args, **kwargs)
     A ProgressBar which knows how to go back to the beginning of the line.
class translate.misc.progressbar.MessageProgressBar(*args, **kwargs)
     A ProgressBar that just writes out the messages without any progress display
class translate.misc.progressbar.NoProgressBar
     An invisible indicator that does nothing.
     show (verbosemessage)
          show nothing for progress :-)
class translate.misc.progressbar.ProgressBar(minValue=0,
                                                                          maxValue=100,
                                                                                            total-
                                                           Width=50)
     A plain progress bar that doesn't know very much about output.
     show (verbosemessage)
          displays the progress bar
```

# quote

String processing utilities for extracting strings with various kinds of delimiters

```
translate.misc.quote.entitydecode (source, name2codepoint)

Decode source using entities from name2codepoint.
```

### **Parameters**

- source (unicode) Source string to decode
- name2codepoint (dict()) Dictionary mapping entity names (without the he leading & or the trailing;) to code points

translate.misc.quote.entityencode (source, codepoint2name)

Encode source using entities from codepoint2name.

### **Parameters**

- source (unicode) Source string to encode
- codepoint2name (dict()) Dictionary mapping code points to entity names (without the the leading & or the trailing;)

```
translate.misc.quote.escapecontrols(source)
     escape control characters in the given string
translate.misc.quote.extract (source, startdelim, enddelim, escape=None, startinstring=False, al-
                                      lowreentry=True)
     Extracts a doublequote-delimited string from a string, allowing for backslash-escaping returns tuple of (quoted
     string with quotes, still in string at end).
translate.misc.quote.extractwithoutquotes (source, startdelim, enddelim, escape=None,
                                                        startinstring=False, includeescapes=True, al-
                                                        lowreentry=True)
     Extracts a doublequote-delimited string from a string, allowing for backslash-escaping includeescapes can also
     be a function that takes the whole escaped string and returns the replaced version.
translate.misc.quote.find all(searchin, substr)
     Returns a list of locations where substroccurs in searchin locations are not allowed to overlap
translate.misc.quote.htmlentitydecode(source)
     Decode source using HTML entities e.g. © -> \mathbb{O}.
          Parameters source (unicode) - Source string to decode
translate.misc.quote.htmlentityencode(source)
     Encode source using HTML entities e.g. © -> ©
          Parameters source (unicode) - Source string to encode
translate.misc.quote.java_utf8_properties_encode(source)
     Encodes source in the escaped-unicode encoding used by java utf-8 properties files.
translate.misc.quote.javapropertiesencode(source)
     Encodes source in the escaped-unicode encoding used by Java .properties files
translate.misc.quote.mozillaescapemarginspaces (source)
     Escape leading and trailing spaces for Mozilla .properties files.
translate.misc.quote.propertiesdecode(source)
     Decodes source from the escaped-unicode encoding used by .properties files.
     Java uses Latin1 by default, and Mozilla uses UTF-8 by default.
     Since the .decode("unicode-escape") routine decodes everything, and we don't want to we reimplemented the
     algorithm from Python Objects/unicode.c in Python and modify it to retain escaped control characters.
wsgi
Wrapper to launch the bundled CherryPy server.
translate.misc.wsgi.launch_server(host, port, app, **kwargs)
     Use cheroot WSGI server, a multithreaded scallable server.
wStringIO
A thin wrapper around BytesIO that accepts and auto-convert non bytes input
class translate.misc.wStringIO.CatchStringOutput (onclose)
     catches the output before it is closed and sends it to an onclose method
     close()
          wrap the underlying close method, to pass the value to onclose before it goes
```

```
slam()
          use this method to force the closing of the stream if it isn't closed yet
class translate.misc.wStringIO.StringIO(buf=")
xml helpers
Helper functions for working with XML.
translate.misc.xml_helpers.getText(node, xml_space='preserve')
     Extracts the plain text content out of the given node.
     This method checks the xml:space attribute of the given node, and takes an optional default to use in case nothing
     is specified in this node.
translate.misc.xml_helpers.getXMLlang(node)
     Gets the xml:lang attribute on node
translate.misc.xml_helpers.getXMLspace(node, default=None)
     Gets the xml:space attribute on node
translate.misc.xml_helpers.namespaced(namespace, name)
     Returns name in Clark notation within the given namespace.
     For example namespaced("source") in an XLIFF document might return:
     {urn:oasis:names:tc:xliff:document:1.1}source
     This is needed throughout lxml.
translate.misc.xml_helpers.normalize_space(text)
     Normalize the given text for implementation of xml:space="default".
translate.misc.xml_helpers.normalize_xml_space (node, xml_space, remove_start=False)
     normalize spaces following the nodes xml:space, or alternatively the given xml_space parameter.
translate.misc.xml_helpers.reindent(elem, level=0, indent=' ', max_level=4, skip=None,
                                             toplevel=True)
     Adjust indentation to match specification.
     Each nested tag is identified by indent string, up to max_level depth, possibly skipping tags listed in skip.
translate.misc.xml_helpers.setXMLlang(node, lang)
     Sets the xml:lang attribute on node
translate.misc.xml helpers.setXMLspace (node, value)
     Sets the xml:space attribute on node
translate.misc.xml_helpers.string_xpath = string()
     Return a non-normalized string in the node subtree
translate.misc.xml_helpers.string_xpath_normalized = normalize-space()
     Return a (space) normalized string in the node subtree
translate.misc.xml_helpers.xml_preserve_ancestors = ancestor-or-self::*[attribute::xml:space
     All ancestors with xml:space='preserve'
translate.misc.xml_helpers.xml_space_ancestors = ancestor-or-self::*/attribute::xml:space
     All xml:space attributes in the ancestors
```

### search

Services for searching and matching of text.

# indexing

Interface for differrent indexing engines for the Translate Toolkit.

### CommonIndexer

base class for interfaces to indexing engines for pootle

```
 \textbf{class} \text{ translate.search.indexing.CommonIndexer.} \textbf{CommonDatabase} (basedir, analyzer=None, create\_allowed=True)
```

Base class for indexing support.

Any real implementation must override most methods of this class.

```
ANALYZER DEFAULT = 6
```

the default analyzer to be used if nothing is configured

```
ANALYZER\_EXACT = 0
```

exact matching: the query string must equal the whole term string

```
ANALYZER_PARTIAL = 2
```

partial matching: a document matches, even if the query string only matches the beginning of the term

```
ANALYZER_TOKENIZE = 4
```

tokenize terms and queries automatically

```
INDEX DIRECTORY NAME = None
```

override this with a string to be used as the name of the indexing directory/file in the filesystem

```
QUERY TYPE = None
```

override this with the query class of the implementation

### begin\_transaction()

begin a transaction

You can group multiple modifications of a database as a transaction. This prevents time-consuming database flushing and helps, if you want that a changeset is committed either completely or not at all. No changes will be written to disk until 'commit\_transaction'. 'cancel\_transaction' can be used to revert an ongoing transaction.

Database types that do not support transactions may silently ignore it.

# cancel\_transaction()

cancel an ongoing transaction

See 'start\_transaction' for details.

#### commit transaction()

Submit the currently ongoing transaction and write changes to disk.

See 'start\_transaction' for details.

# delete\_doc(ident)

Delete the documents returned by a query.

Parameters ident (int | list of tuples | dict | list of dicts | query (e.g. xapian.Query) | list of queries) - [list of] document IDs | dict describing a query | query

# delete\_document\_by\_id (docid)

Delete a specified document.

Parameters docid (int) – the document ID to be deleted

### field analyzers = {}

mapping of field names and analyzers - see set\_field\_analyzers()

flush (optimize=False)

Flush the content of the database - to force changes to be written to disk.

Some databases also support index optimization.

**Parameters** optimize (bool) – should the index be optimized if possible?

# get\_field\_analyzers (fieldnames=None)

Return the analyzer that was mapped to a specific field.

See set\_field\_analyzers() for details.

**Parameters fieldnames** (str | list of str | None) – the analyzer of this field (or all/multiple fields) is requested; leave empty (or None) to request all fields.

**Returns** The analyzer setting of the field - see *CommonDatabase.ANALYZER\_???* or a dict of field names and analyzers

Return type int | dict

# get\_query\_result (query)

return an object containing the results of a query

Parameters query (a query object of the real implementation) - a precompiled query

**Returns** an object that allows access to the results

Return type subclass of CommonEnquire

### index\_document (data)

Add the given data to the database.

**Parameters data** (dict | list of str) - the data to be indexed. A dictionary will be treated as fieldname: value combinations. If the fieldname is None then the value will be interpreted as a plain term or as a list of plain terms. Lists of terms are indexed separately. Lists of strings are treated as plain terms.

# make\_query (args, require\_all=True, analyzer=None)

Create simple queries (strings or field searches) or combine multiple queries (AND/OR).

To specify rules for field searches, you may want to take a look at <code>set\_field\_analyzers()</code>. The parameter 'match\_text\_partial' can override the previously defined default setting.

# **Parameters**

• args (list of queries | single query | str | dict) — queries or search string or description of field query examples:

```
[xapian.Query("foo"), xapian.Query("bar")]
xapian.Query("foo")
```

(continues on next page)

(continued from previous page)

```
"bar" {"foo": "bar", "foobar": "foo"}
```

- require\_all (boolean) boolean operator (True -> AND (default) / False -> OR)
- analyzer (int) (only applicable for 'dict' or 'str') Define query options (partial matching, exact matching, tokenizing, ...) as bitwise combinations of *CommonIndexer.ANALYZER\_???*.

This can override previously defined field analyzer settings.

If analyzer is None (default), then the configured analyzer for the field is used.

**Returns** the combined query

Return type query type of the specific implementation

```
search (query, fieldnames)
```

Return a list of the contents of specified fields for all matches of a query.

#### **Parameters**

- query (a query object of the real implementation) the query to be issued
- **fieldnames** (string | list of strings) the name(s) of a field of the document content

**Returns** a list of dicts containing the specified field(s)

Return type list of dicts

### set field analyzers (field analyzers)

Set the analyzers for different fields of the database documents.

All bitwise combinations of *CommonIndexer.ANALYZER*\_??? are possible.

```
Parameters field_analyzers (dict containing field names and analyzers) – mapping of field names and analyzers
```

Raises TypeError – invalid values in field\_analyzers

class translate.search.indexing.CommonIndexer.CommonEnquire(enquire)

An enquire object contains the information about the result of a request.

```
get_matches (start, number)
```

Return a specified number of qualified matches of a previous query.

#### **Parameters**

- **start** (*int*) index of the first match to return (starting from zero)
- number (int) the number of matching entries to return

**Returns** a set of matching entries and some statistics

# Return type

tuple of (returned number, available number, matches) "matches" is a dictionary of:

```
["rank", "percent", "document", "docid"]
```

# get\_matches\_count()

Return the estimated number of matches.

Use translate.search.indexing.CommonIndexer.search() to retrieve the exact number of matches

**Returns** The estimated number of matches

Return type int

```
translate.search.indexing.CommonIndexer.is_available()
```

Check if this indexing engine interface is usable.

This function must exist in every module that contains indexing engine interfaces.

**Returns** is this interface usable?

Return type bool

# **PyLuceneIndexer**

interface for the PyLucene (v2.x) indexing engine

take a look at PyLuceneIndexer1.py for the PyLucene v1.x interface

```
 \textbf{class} \text{ translate.search.indexing.PyLuceneIndexer.PyLuceneDatabase} (basedir, analyzer=None, \\ cre-\\ ate\_allowed=True)
```

Manage and use a pylucene indexing database.

# begin\_transaction()

PyLucene does not support transactions

Thus this function just opens the database for write access. Call "cancel\_transaction" or "commit\_transaction" to close write access in order to remove the exclusive lock from the database directory.

# cancel\_transaction()

PyLucene does not support transactions

Thus this function just closes the database write access and removes the exclusive lock.

See 'start transaction' for details.

# commit\_transaction()

PyLucene does not support transactions

Thus this function just closes the database write access and removes the exclusive lock.

See 'start transaction' for details.

### delete doc(ident)

Delete the documents returned by a query.

```
Parameters ident (int | list of tuples | dict | list of dicts | query (e.g. xapian.Query) | list of queries) - [list of] document IDs | dict describing a query | query
```

# delete\_document\_by\_id(docid)

delete a specified document

Parameters docid (int) - the document ID to be deleted

```
flush (optimize=False)
```

flush the content of the database - to force changes to be written to disk

some databases also support index optimization

**Parameters** optimize (bool) – should the index be optimized if possible?

```
get field analyzers (fieldnames=None)
```

Return the analyzer that was mapped to a specific field.

```
See set_field_analyzers() for details.
```

**Parameters fieldnames** (str | list of str | None) – the analyzer of this field (or all/multiple fields) is requested; leave empty (or None) to request all fields.

**Returns** The analyzer setting of the field - see *CommonDatabase.ANALYZER\_???* or a dict of field names and analyzers

Return type int | dict

```
get_query_result (query)
```

return an object containing the results of a query

Parameters query (a query object of the real implementation) - a precompiled query

**Returns** an object that allows access to the results

**Return type** subclass of CommonEnquire

```
index document (data)
```

Add the given data to the database.

**Parameters data** (dict | list of str) - the data to be indexed. A dictionary will be treated as fieldname: value combinations. If the fieldname is None then the value will be interpreted as a plain term or as a list of plain terms. Lists of terms are indexed separately. Lists of strings are treated as plain terms.

```
make_query(*args, **kwargs)
```

Create simple queries (strings or field searches) or combine multiple queries (AND/OR).

To specify rules for field searches, you may want to take a look at <code>set\_field\_analyzers()</code>. The parameter 'match\_text\_partial' can override the previously defined default setting.

### **Parameters**

• args (list of queries | single query | str | dict) - queries or search string or description of field query examples:

```
[xapian.Query("foo"), xapian.Query("bar")]
xapian.Query("foo")
"bar"
{"foo": "bar", "foobar": "foo"}
```

- require\_all (boolean) boolean operator (True -> AND (default) / False -> OR)
- **analyzer** (*int*) (only applicable for 'dict' or 'str') Define query options (partial matching, exact matching, tokenizing, ...) as bitwise combinations of *Commonln-dexer.ANALYZER\_???*.

This can override previously defined field analyzer settings.

If analyzer is None (default), then the configured analyzer for the field is used.

**Returns** the combined query

Return type query type of the specific implementation

```
search (query, fieldnames)
```

Return a list of the contents of specified fields for all matches of a query.

#### **Parameters**

- query (a query object of the real implementation) the query to be issued
- **fieldnames** (*string* / *list of strings*) the name(s) of a field of the document content

**Returns** a list of dicts containing the specified field(s)

Return type list of dicts

# set\_field\_analyzers (field\_analyzers)

Set the analyzers for different fields of the database documents.

All bitwise combinations of CommonIndexer.ANALYZER\_??? are possible.

Parameters field\_analyzers (dict containing field names and analyzers) — mapping of field names and analyzers

Raises TypeError – invalid values in field\_analyzers

class translate.search.indexing.PyLuceneIndexer.PyLuceneHits (enquire)
 an enquire object contains the information about the result of a request

# get\_matches (start, number)

return a specified number of qualified matches of a previous query

### **Parameters**

- **start** (*int*) index of the first match to return (starting from zero)
- **number** (int) the number of matching entries to return

Returns a set of matching entries and some statistics

# Return type

tuple of (returned number, available number, matches) "matches" is a dictionary of:

```
["rank", "percent", "document", "docid"]
```

### get\_matches\_count()

Return the estimated number of matches.

Use translate.search.indexing.CommonIndexer.search() to retrieve the exact number of matches

**Returns** The estimated number of matches

Return type int

### **XapianIndexer**

Interface to the Xapian indexing engine for the Translate Toolkit

Xapian v1.0 or higher is supported.

If you are interested in writing an interface for Xapian 0.x, then you should checkout the following:

```
svn export -r 7235 https://translate.svn.sourceforge.net/svnroot/translate/src/

→branches/translate-search-indexer-generic-merging/translate/search/indexer/
```

It is not completely working, but it should give you a good start.

Interface to the Xapian indexer.

### begin\_transaction()

Begin a transaction.

Xapian supports transactions to group multiple database modifications. This avoids intermediate flushing and therefore increases performance.

#### cancel transaction()

cancel an ongoing transaction

no changes since the last execution of 'begin\_transcation' are written

### commit\_transaction()

Submit the changes of an ongoing transaction.

All changes since the last execution of 'begin\_transaction' are written.

# delete\_doc(ident)

Delete the documents returned by a query.

```
Parameters ident (int | list of tuples | dict | list of dicts | query (e.g. xapian.Query) | list of queries) - [list of] document IDs | dict describing a query | query
```

# delete\_document\_by\_id (docid)

Delete a specified document.

Parameters docid (int) – the document ID to be deleted

```
flush (optimize=False)
```

force to write the current changes to disk immediately

Parameters optimize (bool) – ignored for xapian

```
get_field_analyzers (fieldnames=None)
```

Return the analyzer that was mapped to a specific field.

```
See set_field_analyzers() for details.
```

**Parameters fieldnames** (str | list of str | None) – the analyzer of this field (or all/multiple fields) is requested; leave empty (or None) to request all fields.

**Returns** The analyzer setting of the field - see *CommonDatabase.ANALYZER\_???* or a dict of field names and analyzers

Return type int | dict

# get\_query\_result (query)

Return an object containing the results of a query.

Parameters query (xapian.Query) – a pre-compiled xapian query

**Returns** an object that allows access to the results

Return type XapianIndexer.CommonEnquire

### index\_document (data)

Add the given data to the database.

**Parameters data** (dict | list of str) - the data to be indexed. A dictionary will be treated as fieldname: value combinations. If the fieldname is None then the value will

be interpreted as a plain term or as a list of plain terms. Lists of terms are indexed separately. Lists of strings are treated as plain terms.

```
make_query(*args, **kwargs)
```

Create simple queries (strings or field searches) or combine multiple queries (AND/OR).

To specify rules for field searches, you may want to take a look at set\_field\_analyzers(). The parameter 'match\_text\_partial' can override the previously defined default setting.

#### **Parameters**

• args (list of queries | single query | str | dict) - queries or search string or description of field query examples:

```
[xapian.Query("foo"), xapian.Query("bar")]
xapian.Query("foo")
"bar"
{"foo": "bar", "foobar": "foo"}
```

- require\_all (boolean) boolean operator (True -> AND (default) / False -> OR)
- **analyzer** (*int*) (only applicable for 'dict' or 'str') Define query options (partial matching, exact matching, tokenizing, ...) as bitwise combinations of *CommonIndexer.ANALYZER\_???*.

This can override previously defined field analyzer settings.

If analyzer is None (default), then the configured analyzer for the field is used.

**Returns** the combined query

Return type query type of the specific implementation

```
search (query, fieldnames)
```

Return a list of the contents of specified fields for all matches of a query.

#### **Parameters**

- query (xapian. Query) the query to be issued
- **fieldnames** (*string* / *list* of *strings*) the name(s) of a field of the document content

**Returns** a list of dicts containing the specified field(s)

Return type list of dicts

```
set_field_analyzers (field_analyzers)
```

Set the analyzers for different fields of the database documents.

All bitwise combinations of CommonIndexer.ANALYZER\_??? are possible.

```
Parameters field_analyzers (dict containing field names and analyzers) – mapping of field names and analyzers
```

Raises TypeError – invalid values in field\_analyzers

class translate.search.indexing.XapianIndexer.XapianEnquire(enquire)
 interface to the xapian object for storing sets of matches

```
get_matches (start, number)
```

Return a specified number of qualified matches of a previous query.

### **Parameters**

• **start** (*int*) – index of the first match to return (starting from zero)

• **number** (*int*) – the number of matching entries to return

Returns a set of matching entries and some statistics

# Return type

tuple of (returned number, available number, matches) "matches" is a dictionary of:

```
["rank", "percent", "document", "docid"]
```

```
get_matches_count()
```

Return the estimated number of matches.

Use translate.search.indexing.CommonIndexer.search() to retrieve the exact number of matches

**Returns** The estimated number of matches

Return type int

### Ishtein

A class to calculate a similarity based on the Levenshtein distance.

See http://en.wikipedia.org/wiki/Levenshtein\_distance.

If available, the python-Levenshtein will be used which will provide better performance as it is implemented natively.

```
translate.search.lshtein.distance(a, b, stopvalue=0)
```

Same as python\_distance in functionality. This uses the fast C version if we detected it earlier.

Note that this does not support arbitrary sequence types, but only string types.

```
translate.search.lshtein.native_distance(a, b, stopvalue=0)
```

Same as python\_distance in functionality. This uses the fast C version if we detected it earlier.

Note that this does not support arbitrary sequence types, but only string types.

```
translate.search.lshtein.python distance(a, b, stopvalue=-1)
```

Calculates the distance for use in similarity calculation. Python version.

### match

Class to perform translation memory matching from a store of translation units.

```
 \textbf{class} \text{ translate.search.match.matcher} (\textit{store}, \textit{max\_candidates=10}, \textit{min\_similarity=75}, \\ \textit{max\_length=70}, \textit{comparer=None}, \textit{usefuzzy=False})
```

A class that will do matching and store configuration for the matching process.

```
buildunits (candidates)
```

Builds a list of units conforming to base API, with the score in the comment.

```
extendtm (units, store=None, sort=True)
```

Extends the memory with extra unit(s).

### **Parameters**

- units The units to add to the TM.
- **store** Optional store from where some metadata can be retrieved and associated with each unit.

• **sort** – Optional parameter that can be set to False to supress sorting of the candidates list. This should probably only be used in *matcher.inittm()*.

# getstartlength (min\_similarity, text)

Calculates the minimum length we are interested in. The extra fat is because we don't use plain character distance only.

# getstoplength (min\_similarity, text)

Calculates a length beyond which we are not interested. The extra fat is because we don't use plain character distance only.

### inittm(stores, reverse=False)

Initialises the memory for later use. We use simple base units for speedup.

#### matches (text)

Returns a list of possible matches for given source text.

**Parameters** text (String) – The text that will be search for in the translation memory

Return type list

**Returns** a list of units with the source and target strings from the translation memory. If self. addpercentage is *True* (default) the match quality is given as a percentage in the notes.

```
setparameters (max_candidates=10, min_similarity=75, max_length=70)
```

Sets the parameters without reinitialising the tm. If a parameter is not specified, it is set to the default, not ignored

#### usable (unit)

Returns whether this translation unit is usable for TM

```
translate.search.match.sourcelen(unit)
```

Returns the length of the source string.

A matcher with settings specifically for terminology matching.

# buildunits (candidates)

Builds a list of units conforming to base API, with the score in the comment.

```
extendtm (units, store=None, sort=True)
```

Extends the memory with extra unit(s).

### **Parameters**

- units The units to add to the TM.
- store Optional store from where some metadata can be retrieved and associated with each unit.
- **sort** Optional parameter that can be set to False to supress sorting of the candidates list. This should probably only be used in *matcher.inittm()*.

# getstartlength (min\_similarity, text)

Calculates the minimum length we are interested in. The extra fat is because we don't use plain character distance only.

### getstoplength (min\_similarity, text)

Calculates a length beyond which we are not interested. The extra fat is because we don't use plain character distance only.

```
inittm(store)
```

Normal initialisation, but convert all source strings to lower case

```
matches (text)
```

Normal matching after converting text to lower case. Then replace with the original unit to retain comments, etc.

```
setparameters (max_candidates=10, min_similarity=75, max_length=70)
```

Sets the parameters without reinitialising the tm. If a parameter is not specified, it is set to the default, not ignored

# usable (unit)

Returns whether this translation unit is usable for terminology.

```
translate.search.match.unit2dict(unit)
```

converts a pounit to a simple dict structure for use over the web

# segment

Module to deal with different types and uses of segmentation

```
\verb|translate.search.segment.character_iter| (\textit{text})
```

Returns an iterator over the characters in text.

```
translate.search.segment.characters(text)
```

Returns a list of characters in text.

translate.search.segment.sentence\_iter(text)

Returns an iterator over the senteces in text.

```
translate.search.segment.sentences(text)
```

Returns a list of senteces in text.

```
translate.search.segment.word_iter(text)
```

Returns an iterator over the words in text.

```
translate.search.segment.words(text)
```

Returns a list of words in text.

# terminology

A class that does terminology matching

### services

translate.services is part of the translate toolkit. It provides network services for interacting with the toolkit

# tmserver

A translation memory server using tmdb for storage, communicates with clients using JSON over HTTP.

```
class translate.services.tmserver. TMServer (tmdbfile, tmfiles, max_candidates=3, min_similarity=75, max_length=1000, prefix=", source_lang=None, target_lang=None) target_lang=None
```

A RESTful JSON TM server.

### storage

Classes that represent various storage formats for localization.

#### base

Base classes for storage interfaces.

```
exception translate.storage.base.ParseError(inner_exc)
```

**class** translate.storage.base.**TranslationStore**(unitclass=None, encoding=None)

Base class for stores for multiple translation units of type UnitClass.

#### Extensions = None

A list of file extentions associated with this store type

# Mimetypes = None

A list of MIME types associated with this store type

### Name = 'Base translation store'

The human usable name of this store type

### UnitClass

The class of units that will be instantiated and used by this class

alias of TranslationUnit

#### add\_unit\_to\_index(unit)

Add a unit to source and location idexes

# addsourceunit (source)

Add and returns a new unit with the given source string.

Return type TranslationUnit

# addunit (unit)

Append the given unit to the object's list of units.

This method should always be used rather than trying to modify the list manually.

**Parameters unit** (*TranslationUnit*) – The unit that will be added.

# detect\_encoding(text, default\_encodings=None)

Try to detect a file encoding from text, using either the chardet lib or by trying to decode the file.

# $fallback\_detection(text)$

Simple detection based on BOM in case chardet is not available.

### findid(id)

find unit with matching id by checking id\_index

# findunit (source)

Find the unit with the given source string.

Return type TranslationUnit or None

#### findunits (source)

Find the units with the given source string.

Return type TranslationUnit or None

# getids (filename=None)

return a list of unit ids

# getprojectstyle() Get the project type for this store. getsourcelanguage() Get the source language for this store. gettargetlanguage() Get the target language for this store. getunits() Return a list of all units in this store. isempty() Return True if the object doesn't contain any translation units. makeindex() Indexes the items in this store. At least .sourceindex should be useful. merge\_on The matching criterion to use when merging on. **Returns** The default matching criterion for all the subclasses. **Return type** string parse (data) parser to process the given source string classmethod parsefile (storefile) Reads the given file (or opens the given filename) and parses back to an object. classmethod parsestring(storestring) Convert the string representation back to an object. remove\_unit\_from\_index(unit) Remove a unit from source and locaton indexes require\_index() make sure source index exists save() Save to the file that data was originally read from, if available. savefile (storefile) Write the string representation to the given file (or filename). serialize(out) Converts to a bytes representation that can be parsed back using parsestring(). out should be an open file-like objects to write to. setprojectstyle (project\_style) Set the project type for this store. setsourcelanguage (sourcelanguage) Set the source language for this store. settargetlanguage (targetlanguage) Set the target language for this store.

4.1. API 409

Indicates if format can store suggestions and alternative translation for a unit

suggestions\_in\_format = False

Return the translated string for a given source string.

translate(source)

### **Return type** String or None

### unit iter()

Iterator over all the units in this store.

class translate.storage.base.TranslationUnit(source=None)

Base class for translation units.

Our concept of a translation unit is influenced heavily by XLIFF.

As such most of the method- and variable names borrows from XLIFF terminology.

A translation unit consists of the following:

- A source string. This is the original translatable text.
- A *target* string. This is the translation of the *source*.
- Zero or more *notes* on the unit. Notes would typically be some comments from a translator on the unit, or some comments originating from the source code.
- Zero or more *locations*. Locations indicate where in the original source code this unit came from.
- Zero or more *errors*. Some tools (eg. *pofilter*) can run checks on translations and produce error messages.

# adderror (errorname, errortext)

Adds an error message to this unit.

#### **Parameters**

- errorname (string) A single word to id the error.
- **errortext** (*string*) The text describing the error.

# addlocation (location)

Add one location to the list of locations.

Note: Shouldn't be implemented if the format doesn't support it.

# addlocations (location)

Add a location or a list of locations.

**Note:** Most classes shouldn't need to implement this, but should rather implement *TranslationUnit.addlocation()*.

**Warning:** This method might be removed in future.

addnote (text, origin=None, position='append')

Adds a note (comment).

### **Parameters**

- text (string) Usually just a sentence or two.
- **origin** (*string*) Specifies who/where the comment comes from. Origin can be one of the following text strings: 'translator' 'developer', 'programmer', 'source code' (synonyms)

### classmethod buildfromunit (unit)

Build a native unit from a foreign unit, preserving as much information as possible.

# getcontext()

Get the message context.

### geterrors()

Get all error messages.

### **Return type** Dictionary

### getid()

A unique identifier for this unit.

### Return type string

**Returns** an identifier for this unit that is unique in the store

Derived classes should override this in a way that guarantees a unique identifier for each unit in the store.

### getlocations()

A list of source code locations.

### **Return type** List

**Note:** Shouldn't be implemented if the format doesn't support it.

### getnotes (origin=None)

Returns all notes about this unit.

It will probably be freeform text or something reasonable that can be synthesised by the format. It should not include location comments (see <code>getlocations()</code>).

# gettargetlen()

Returns the length of the target string.

# Return type Integer

**Note:** Plural forms might be combined.

# getunits()

This unit in a list.

# hasplural()

Tells whether or not this specific unit has plural strings.

# infer\_state()

Empty method that should be overridden in sub-classes to infer the current state(\_n) of the unit from its current state.

### isblank()

Used to see if this unit has no source or target string.

**Note:** This is probably used more to find translatable units, and we might want to move in that direction rather and get rid of this.

# isfuzzy()

Indicates whether this unit is fuzzy.

#### isheader()

Indicates whether this unit is a header.

# isobsolete()

indicate whether a unit is obsolete

### isreview()

Indicates whether this unit needs review.

#### istranslatable()

Indicates whether this unit can be translated.

This should be used to distinguish real units for translation from header, obsolete, binary or other blank units.

#### istranslated()

Indicates whether this unit is translated.

This should be used rather than deducing it from .target, to ensure that other classes can implement more functionality (as XLIFF does).

### makeobsolete()

Make a unit obsolete

# markfuzzy(value=True)

Marks the unit as fuzzy or not.

# markreviewneeded (needsreview=True, explanation=None)

Marks the unit to indicate whether it needs review.

#### **Parameters**

- needsreview Defaults to True.
- **explanation** Adds an optional explanation as a note.

 $\textbf{merge} \ (other unit, overwrite = False, comments = True, authoritative = False)$ 

Do basic format agnostic merging.

### multistring\_to\_rich(mulstring)

Convert a multistring to a list of "rich" string trees:

# removenotes (origin=None)

Remove all the translator's notes.

#### rich parsers = []

A list of functions to use for parsing a string into a rich string tree.

# rich\_source

### See also:

```
rich_to_multistring(), multistring_to_rich()
```

# rich\_target

# See also:

```
rich_to_multistring(), multistring_to_rich()
     classmethod rich_to_multistring(elem_list)
          Convert a "rich" string tree to a multistring:
          >>> from translate.storage.placeables.interfaces import X
          >>> rich = [StringElem(['foo', X(id='xxx', sub=[' ']), 'bar'])]
          >>> TranslationUnit.rich_to_multistring(rich)
          multistring(u'foo bar')
     setcontext (context)
          Set the message context
     setid(value)
          Sets the unique identified for this unit.
          only implemented if format allows ids independant from other unit properties like source or context
     unit_iter()
          Iterator that only returns this unit.
benchmark
class translate.storage.benchmark.TranslateBenchmarker(test_dir, storeclass)
     class to aid in benchmarking Translate Toolkit stores
     clear_test_dir()
          removes the given directory
     create_sample_files (num_dirs, files_per_dir, strings_per_file, source_words_per_string, tar-
                               get_words_per_string)
          creates sample files for benchmarking
     parse_files (file_dir=None)
          parses all the files in the test directory into memory
     parse_placeables()
          parses placeables
bundleprojstore
class translate.storage.bundleprojstore.BundleProjectStore (fname)
     Represents a translate project bundle (zip archive).
     append_file (afile, fname, ftype='trans', delete_orig=False)
          Append the given file to the project with the given filename, marked to be of type ftype ('src', 'trans',
          'tgt').
              Parameters delete_orig - If True, as set by convert_forward(), afile is deleted
                  after appending, if possible.
```

Clean up our mess: remove temporary files.

cleanup()

4.1. API 413

**Note:** For this implementation, the appended file will be deleted from disk if delete orig is True.

```
get file(fname)
     Retrieve a project file (source, translation or target file) from the project archive.
get_filename_type (fname)
     Get the type of file ('src', 'trans', 'tgt') with the given name.
get proj filename(realfname)
     Try and find a project file name for the given real file name.
load (zipname)
     Load the bundle project from the zip file of the given name.
remove_file (fname, ftype=None)
     Remove the file with the given project name from the project.
save (filename=None)
     Save all project files to the bundle zip file.
sourcefiles
     Read-only access to self._sourcefiles.
targetfiles
     Read-only access to self._targetfiles.
transfiles
     Read-only access to self._transfiles.
update file (pfname, infile)
     Updates the file with the given project file name with the contents of infile.
```

exception translate.storage.bundleprojstore.InvalidBundleError

### catkeys

Manage the Haiku catkeys translation format

The Haiku catkeys format is the translation format used for localisation of the Haiku operating system.

**Returns** the results from BundleProjStore.append file().

It is a bilingual base class derived format with <code>CatkeysFile</code> and <code>CatkeysUnit</code> providing file and unit level access. The file format is described here: <a href="http://www.haiku-os.org/blog/pulkomandy/2009-09-24\_haiku\_locale\_kit\_translator\_handbook">http://www.haiku-os.org/blog/pulkomandy/2009-09-24\_haiku\_locale\_kit\_translator\_handbook</a>

**Implementation** The implementation covers the full requirements of a catkeys file. The files are simple Tab Separated Value (TSV) files that can be read by Microsoft Excel and other spreadsheet programs. They use the .txt extension which does make it more difficult to automatically identify such files.

The dialect of the TSV files is specified by CatkeysDialect.

**Encoding** The files are UTF-8 encoded.

Header CatkeysHeader provides header management support.

**Escaping** catkeys seem to escape things like in C++ (strings are just extracted from the source code unchanged, it seems.

```
Functions allow for _escape() and _unescape().
```

```
class translate.storage.catkeys.CatkeysDialect
```

Describe the properties of a catkeys generated TAB-delimited file.

# classmethod combine (dialect, fmtparams) Create a new dialect with defaults and added parameters. class translate.storage.catkeys.CatkeysFile(inputfile=None, \*\*kwargs) A catkeys translation memory file UnitClass alias of CatkeysUnit add unit to index (unit) Add a unit to source and location idexes addsourceunit (source) Add and returns a new unit with the given source string. Return type TranslationUnit addunit (unit) Append the given unit to the object's list of units. This method should always be used rather than trying to modify the list manually. Parameters unit (TranslationUnit) - The unit that will be added. detect\_encoding(text, default\_encodings=None) Try to detect a file encoding from *text*, using either the chardet lib or by trying to decode the file. fallback\_detection (text) Simple detection based on BOM in case chardet is not available. findid(id) find unit with matching id by checking id\_index findunit (source) Find the unit with the given source string. Return type TranslationUnit or None findunits (source) Find the units with the given source string. Return type TranslationUnit or None getids (filename=None) return a list of unit ids getprojectstyle() Get the project type for this store. getsourcelanguage() Get the source language for this store. gettargetlanguage() Get the target language for this store. getunits() Return a list of all units in this store.

4.1. API 415

Return True if the object doesn't contain any translation units.

Indexes the items in this store. At least .sourceindex should be useful.

makeindex()

# merge\_on

The matching criterion to use when merging on.

**Returns** The default matching criterion for all the subclasses.

Return type string

#### parse (input)

parse the given file or file source string

# classmethod parsefile(storefile)

Reads the given file (or opens the given filename) and parses back to an object.

### classmethod parsestring(storestring)

Convert the string representation back to an object.

# remove\_unit\_from\_index(unit)

Remove a unit from source and locaton indexes

## require\_index()

make sure source index exists

#### save()

Save to the file that data was originally read from, if available.

### savefile (storefile)

Write the string representation to the given file (or filename).

#### serialize (out)

Converts to a bytes representation that can be parsed back using parsestring(). out should be an open file-like objects to write to.

# setprojectstyle (project\_style)

Set the project type for this store.

# setsourcelanguage (sourcelanguage)

Set the source language for this store.

# settargetlanguage (newlang)

Set the target language for this store.

### translate(source)

Return the translated string for a given source string.

Return type String or None

### unit iter()

Iterator over all the units in this store.

# class translate.storage.catkeys.CatkeysHeader(header=None)

A catkeys translation memory header

# settargetlanguage (newlang)

Set a human readable target language

# class translate.storage.catkeys.CatkeysUnit(source=None)

A catkeys translation memory unit

### adderror (errorname, errortext)

Adds an error message to this unit.

# **Parameters**

• **errorname** (*string*) – A single word to id the error.

• **errortext** (*string*) – The text describing the error.

### addlocation (location)

Add one location to the list of locations.

**Note:** Shouldn't be implemented if the format doesn't support it.

### addlocations (location)

Add a location or a list of locations.

**Note:** Most classes shouldn't need to implement this, but should rather implement TranslationUnit.addlocation().

**Warning:** This method might be removed in future.

# addnote (text, origin=None, position='append')

Adds a note (comment).

### **Parameters**

- **text** (*string*) Usually just a sentence or two.
- **origin** (*string*) Specifies who/where the comment comes from. Origin can be one of the following text strings: 'translator' 'developer', 'programmer', 'source code' (synonyms)

# classmethod buildfromunit (unit)

Build a native unit from a foreign unit, preserving as much information as possible.

# dict

Get the dictionary of values for a catkeys line

# getcontext()

Get the message context.

# getdict()

Get the dictionary of values for a catkeys line

# geterrors()

Get all error messages.

Return type Dictionary

# getid()

A unique identifier for this unit.

### **Return type** string

Returns an identifier for this unit that is unique in the store

Derived classes should override this in a way that guarantees a unique identifier for each unit in the store.

### getlocations()

A list of source code locations.

# Return type List

Note: Shouldn't be implemented if the format doesn't support it.

# getnotes (origin=None)

Returns all notes about this unit.

It will probably be freeform text or something reasonable that can be synthesised by the format. It should not include location comments (see <code>getlocations()</code>).

# gettargetlen()

Returns the length of the target string.

### Return type Integer

**Note:** Plural forms might be combined.

### getunits()

This unit in a list.

### hasplural()

Tells whether or not this specific unit has plural strings.

#### infer state()

Empty method that should be overridden in sub-classes to infer the current state(\_n) of the unit from its current state.

#### isblank()

Used to see if this unit has no source or target string.

**Note:** This is probably used more to find translatable units, and we might want to move in that direction rather and get rid of this.

# isfuzzy()

Indicates whether this unit is fuzzy.

### isheader()

Indicates whether this unit is a header.

# isobsolete()

indicate whether a unit is obsolete

### isreview()

Indicates whether this unit needs review.

### istranslatable()

Indicates whether this unit can be translated.

This should be used to distinguish real units for translation from header, obsolete, binary or other blank units.

#### istranslated()

Indicates whether this unit is translated.

This should be used rather than deducing it from .target, to ensure that other classes can implement more functionality (as XLIFF does).

# makeobsolete()

Make a unit obsolete

```
markfuzzy (present=True)
```

Marks the unit as fuzzy or not.

### markreviewneeded (needsreview=True, explanation=None)

Marks the unit to indicate whether it needs review.

### **Parameters**

- needsreview Defaults to True.
- **explanation** Adds an optional explanation as a note.

merge (otherunit, overwrite=False, comments=True, authoritative=False)

Do basic format agnostic merging.

# multistring\_to\_rich (mulstring)

Convert a multistring to a list of "rich" string trees:

# removenotes (origin=None)

Remove all the translator's notes.

### rich\_source

### See also:

```
rich_to_multistring(), multistring_to_rich()
```

# rich\_target

# See also:

```
rich_to_multistring(), multistring_to_rich()
```

# classmethod rich\_to\_multistring(elem\_list)

Convert a "rich" string tree to a multistring:

```
>>> from translate.storage.placeables.interfaces import X
>>> rich = [StringElem(['foo', X(id='xxx', sub=[' ']), 'bar'])]
>>> TranslationUnit.rich_to_multistring(rich)
multistring(u'foo bar')
```

# setcontext (context)

Set the message context

### setdict (newdict)

Set the dictionary of values for a catkeys line

**Parameters** newdict (Dict) – a new dictionary with catkeys line elements

### setid(value)

Sets the unique identified for this unit.

only implemented if format allows ids independant from other unit properties like source or context

### unit\_iter()

Iterator that only returns this unit.

```
translate.storage.catkeys.FIELDNAMES = [u'source', u'context', u'comment', u'target']
     Field names for a catkeys TU
translate.storage.catkeys.FIELDNAMES_HEADER = [u'version', u'language', u'mimetype', u'che
     Field names for the catkeys header
translate.storage.catkeys.FIELDNAMES_HEADER_DEFAULTS = {u'checksum': u'', u'language': u
     Default or minimum header entries for a catkeys file
сро
csvl10n
classes that hold units of comma-separated values (.csv) files (csvunit) or entire files (csvfile) for use with localisation
class translate.storage.csvl10n.csvfile(inputfile=None,
                                                                       fieldnames=None,
                                                                                            encod-
                                                     ing=u'auto')
     This class represents a .csv file with various lines. The default format contains three columns: location, source,
     target
     UnitClass
          alias of csvunit
     add_unit_to_index(unit)
          Add a unit to source and location idexes
     addsourceunit (source)
          Add and returns a new unit with the given source string.
              Return type TranslationUnit
     addunit (unit)
          Append the given unit to the object's list of units.
          This method should always be used rather than trying to modify the list manually.
              Parameters unit (TranslationUnit) - The unit that will be added.
     detect_encoding (text, default_encodings=None)
          Try to detect a file encoding from text, using either the chardet lib or by trying to decode the file.
     fallback_detection (text)
          Simple detection based on BOM in case chardet is not available.
          find unit with matching id by checking id_index
     findunit (source)
          Find the unit with the given source string.
              Return type TranslationUnit or None
     findunits (source)
          Find the units with the given source string.
              Return type TranslationUnit or None
     getids (filename=None)
          return a list of unit ids
     getprojectstyle()
          Get the project type for this store.
```

```
getsourcelanguage()
          Get the source language for this store.
     gettargetlanguage()
           Get the target language for this store.
     getunits()
          Return a list of all units in this store.
     isempty()
          Return True if the object doesn't contain any translation units.
     makeindex()
           Indexes the items in this store. At least .sourceindex should be useful.
     merge_on
           The matching criterion to use when merging on.
               Returns The default matching criterion for all the subclasses.
               Return type string
     parse (csvsrc)
           parser to process the given source string
     classmethod parsefile(storefile)
           Reads the given file (or opens the given filename) and parses back to an object.
     classmethod parsestring(storestring)
           Convert the string representation back to an object.
     remove_unit_from_index(unit)
           Remove a unit from source and locaton indexes
     require_index()
           make sure source index exists
     save()
           Save to the file that data was originally read from, if available.
     savefile (storefile)
           Write the string representation to the given file (or filename).
     serialize (out)
           Write to file
     setprojectstyle (project_style)
          Set the project type for this store.
     setsourcelanguage (sourcelanguage)
           Set the source language for this store.
     settargetlanguage (targetlanguage)
           Set the target language for this store.
     translate(source)
           Return the translated string for a given source string.
               Return type String or None
     unit iter()
           Iterator over all the units in this store.
class translate.storage.csvl10n.csvunit(source=None)
```

### add\_spreadsheet\_escapes (source, target)

add common spreadsheet escapes to two strings

# adderror (errorname, errortext)

Adds an error message to this unit.

### **Parameters**

- **errorname** (*string*) A single word to id the error.
- **errortext** (*string*) The text describing the error.

### addlocation (location)

Add one location to the list of locations.

**Note:** Shouldn't be implemented if the format doesn't support it.

### addlocations (location)

Add a location or a list of locations.

**Note:** Most classes shouldn't need to implement this, but should rather implement TranslationUnit.addlocation().

Warning: This method might be removed in future.

# addnote (text, origin=None, position=u'append')

Adds a note (comment).

# **Parameters**

- **text** (*string*) Usually just a sentence or two.
- **origin** (*string*) Specifies who/where the comment comes from. Origin can be one of the following text strings: 'translator' 'developer', 'programmer', 'source code' (synonyms)

### classmethod buildfromunit (unit)

Build a native unit from a foreign unit, preserving as much information as possible.

# getcontext()

Get the message context.

# geterrors()

Get all error messages.

### **Return type** Dictionary

### getid()

A unique identifier for this unit.

# Return type string

**Returns** an identifier for this unit that is unique in the store

Derived classes should override this in a way that guarantees a unique identifier for each unit in the store.

### getlocations()

A list of source code locations.

### **Return type** List

**Note:** Shouldn't be implemented if the format doesn't support it.

# getnotes (origin=None)

Returns all notes about this unit.

It will probably be freeform text or something reasonable that can be synthesised by the format. It should not include location comments (see <code>getlocations()</code>).

# gettargetlen()

Returns the length of the target string.

# Return type Integer

Note: Plural forms might be combined.

### getunits()

This unit in a list.

# hasplural()

Tells whether or not this specific unit has plural strings.

#### infer state()

Empty method that should be overridden in sub-classes to infer the current state(\_n) of the unit from its current state.

# isblank()

Used to see if this unit has no source or target string.

**Note:** This is probably used more to find translatable units, and we might want to move in that direction rather and get rid of this.

# isfuzzy()

Indicates whether this unit is fuzzy.

# isheader()

Indicates whether this unit is a header.

# $\verb|isobsolete|()|$

indicate whether a unit is obsolete

### isreview()

Indicates whether this unit needs review.

# istranslatable()

Indicates whether this unit can be translated.

This should be used to distinguish real units for translation from header, obsolete, binary or other blank units.

### istranslated()

Indicates whether this unit is translated.

This should be used rather than deducing it from .target, to ensure that other classes can implement more functionality (as XLIFF does).

```
makeobsolete()
    Make a unit obsolete
markfuzzy(value=True)
    Marks the unit as fuzzy or not.
markreviewneeded (needsreview=True, explanation=None)
    Marks the unit to indicate whether it needs review.
        Parameters
            • needsreview - Defaults to True.
            • explanation – Adds an optional explanation as a note.
match_header()
    see if unit might be a header
merge (otherunit, overwrite=False, comments=True, authoritative=False)
    Do basic format agnostic merging.
multistring_to_rich(mulstring)
    Convert a multistring to a list of "rich" string trees:
    >>> target = multistring([u'foo', u'bar', u'baz'])
    >>> TranslationUnit.multistring_to_rich(target)
    [<StringElem([<StringElem([u'foo'])>])>,
     <StringElem([<StringElem([u'bar'])>])>,
     <StringElem([<StringElem([u'baz'])>])>]
remove spreadsheet escapes (source, target)
    remove common spreadsheet escapes from two strings
removenotes (origin=None)
    Remove all the translator's notes.
rich source
    See also:
    rich_to_multistring(), multistring_to_rich()
rich_target
    See also:
    rich_to_multistring(), multistring_to_rich()
classmethod rich_to_multistring(elem_list)
    Convert a "rich" string tree to a multistring:
    >>> from translate.storage.placeables.interfaces import X
    >>> rich = [StringElem(['foo', X(id='xxx', sub=[' ']), 'bar'])]
    >>> TranslationUnit.rich_to_multistring(rich)
    multistring(u'foo bar')
```

```
setcontext(value)
```

Set the message context

```
setid(value)
```

Sets the unique identified for this unit.

only implemented if format allows ids independant from other unit properties like source or context

```
unit iter()
```

Iterator that only returns this unit.

```
translate.storage.csvl10n.detect_header(sample, dialect, fieldnames)
```

Test if file has a header or not, also returns number of columns in first row

```
translate.storage.csvl10n.valid_fieldnames(fieldnames)
```

Check if fieldnames are valid, that is at least one field is identified as the source.

### directory

This module provides functionality to work with directories.

```
class translate.storage.directory.Directory(dir=None)
    This class represents a directory.

file_iter()
    Iterator over (dir, filename) for all files in this directory.

getfiles()
    Returns a list of (dir, filename) tuples for all the file names in this directory.

getunits()
    List of all the units in all the files in this directory.

scanfiles()
    Populate the internal file data.
```

# dtd

Classes that hold units of .dtd files (dtdunit) or entire files (dtdfile).

Iterator over all the units in all the files in this directory.

These are specific .dtd files for localisation used by mozilla.

**Specifications** The following information is provided by Mozilla:

# Specification

unit\_iter()

There is a grammar for entity definitions, which isn't really precise, as the spec says. There's no formal specification for DTD files, it's just "whatever makes this work" basically. The whole piece is clearly not the strongest point of the xml spec

XML elements are allowed in entity values. A number of things that are allowed will just break the resulting document, Mozilla forbids these in their DTD parser.

**Dialects** There are two dialects:

- · Regular DTD
- Android DTD

Both dialects are similar, but the Android DTD uses some particular escapes that regular DTDs don't have.

**Escaping in regular DTD** In DTD usually there are characters escaped in the entities. In order to ease the translation some of those escaped characters are unescaped when reading from, or converting, the DTD, and that are escaped again when saving, or converting to a DTD.

In regular DTD the following characters are usually or sometimes escaped:

- The % character is escaped using % or % or %
- The "character is escaped using "
- The 'character is escaped using ' (partial roundtrip)
- The & character is escaped using & amp;
- The < character is escaped using &lt; (not yet implemented)
- The > character is escaped using > (not yet implemented)

Besides the previous ones there are a lot of escapes for a huge number of characters. This escapes usually have the form of &#NUMBER; where NUMBER represents the numerical code for the character.

There are a few particularities in DTD escaping. Some of the escapes are not yet implemented since they are not really necessary, or because its implementation is too hard.

A special case is the 'escaping using ' which doesn't provide a full roundtrip conversion in order to support some special Mozilla DTD files.

Also the "character is never escaped in the case that the previous character is = (the sequence =" is present on the string) in order to avoid escaping the "character indicating an attribute assignment, for example in a href attribute for an a tag in HTML (anchor tag).

**Escaping in Android DTD** It has the sames escapes as in regular DTD, plus this ones:

- The 'character is escaped using ' or 'or u0027
- The "character is escaped using "

translate.storage.dtd.accesskeysuffixes = ('.accesskey', '.accessKey', '.akey')

Accesskey Suffixes: entries with this suffix may be combined with labels ending in labelsuffixes into accelerator notation

class translate.storage.dtd.dtdfile(inputfile=None, android=False)
 A .dtd file made up of dtdunits.

#### UnitClass

alias of dtdunit

# add\_unit\_to\_index(unit)

Add a unit to source and location idexes

# addsourceunit (source)

Add and returns a new unit with the given source string.

Return type TranslationUnit

### addunit (unit)

Append the given unit to the object's list of units.

This method should always be used rather than trying to modify the list manually.

Parameters unit (TranslationUnit) - The unit that will be added.

### detect\_encoding (text, default\_encodings=None)

Try to detect a file encoding from *text*, using either the chardet lib or by trying to decode the file.

# fallback\_detection (text)

Simple detection based on BOM in case chardet is not available.

# findid(id)

find unit with matching id by checking id\_index

```
findunit (source)
     Find the unit with the given source string.
         Return type TranslationUnit or None
findunits (source)
     Find the units with the given source string.
         Return type TranslationUnit or None
getids (filename=None)
     return a list of unit ids
getprojectstyle()
     Get the project type for this store.
getsourcelanguage()
     Get the source language for this store.
gettargetlanguage()
     Get the target language for this store.
getunits()
     Return a list of all units in this store.
isempty()
     Return True if the object doesn't contain any translation units.
makeindex()
     makes self.id_index dictionary keyed on entities
merge_on
     The matching criterion to use when merging on.
         Returns The default matching criterion for all the subclasses.
         Return type string
parse (dtdsrc)
     read the source code of a dtd file in and include them as dtdunits in self.units
classmethod parsefile(storefile)
     Reads the given file (or opens the given filename) and parses back to an object.
classmethod parsestring(storestring)
     Convert the string representation back to an object.
remove_unit_from_index(unit)
     Remove a unit from source and locaton indexes
require_index()
     make sure source index exists
save()
     Save to the file that data was originally read from, if available.
savefile (storefile)
     Write the string representation to the given file (or filename).
serialize(out)
     Write content to file
setprojectstyle (project_style)
```

4.1. API 427

Set the project type for this store.

### setsourcelanguage (sourcelanguage)

Set the source language for this store.

# settargetlanguage (targetlanguage)

Set the target language for this store.

#### translate(source)

Return the translated string for a given source string.

Return type String or None

# unit\_iter()

Iterator over all the units in this store.

### **class** translate.storage.dtd.**dtdunit** (source=", android=False)

An entity definition from a DTD file (and any associated comments).

# adderror (errorname, errortext)

Adds an error message to this unit.

#### **Parameters**

- **errorname** (*string*) A single word to id the error.
- **errortext** (*string*) The text describing the error.

### addlocation (location)

Set the entity to the given "location".

### addlocations (location)

Add a location or a list of locations.

**Note:** Most classes shouldn't need to implement this, but should rather implement TranslationUnit.addlocation().

Warning: This method might be removed in future.

addnote (text, origin=None, position='append')

Adds a note (comment).

### **Parameters**

- **text** (*string*) Usually just a sentence or two.
- **origin** (*string*) Specifies who/where the comment comes from. Origin can be one of the following text strings: 'translator' 'developer', 'programmer', 'source code' (synonyms)

### classmethod buildfromunit (unit)

Build a native unit from a foreign unit, preserving as much information as possible.

# getcontext()

Get the message context.

### geterrors()

Get all error messages.

# Return type Dictionary

# getid()

A unique identifier for this unit.

#### Return type string

Returns an identifier for this unit that is unique in the store

Derived classes should override this in a way that guarantees a unique identifier for each unit in the store.

# getlocations()

Return the entity as location (identifier).

#### getnotes (origin=None)

Returns all notes about this unit.

It will probably be freeform text or something reasonable that can be synthesised by the format. It should not include location comments (see getlocations()).

#### getoutput()

convert the dtd entity back to string form

# gettargetlen()

Returns the length of the target string.

# Return type Integer

Note: Plural forms might be combined.

#### getunits()

This unit in a list.

#### hasplural()

Tells whether or not this specific unit has plural strings.

# infer\_state()

Empty method that should be overridden in sub-classes to infer the current state(\_n) of the unit from its current state.

#### isblank()

Used to see if this unit has no source or target string.

**Note:** This is probably used more to find translatable units, and we might want to move in that direction rather and get rid of this.

# isfuzzy()

Indicates whether this unit is fuzzy.

#### isheader()

Indicates whether this unit is a header.

# isnull()

returns whether this dtdunit doesn't actually have an entity definition

### isobsolete()

indicate whether a unit is obsolete

#### isreview()

Indicates whether this unit needs review.

# istranslatable()

Indicates whether this unit can be translated.

This should be used to distinguish real units for translation from header, obsolete, binary or other blank units.

#### istranslated()

Indicates whether this unit is translated.

This should be used rather than deducing it from .target, to ensure that other classes can implement more functionality (as XLIFF does).

#### makeobsolete()

Make a unit obsolete

# markfuzzy(value=True)

Marks the unit as fuzzy or not.

# markreviewneeded (needsreview=True, explanation=None)

Marks the unit to indicate whether it needs review.

#### **Parameters**

- needsreview Defaults to True.
- **explanation** Adds an optional explanation as a note.

```
merge (otherunit, overwrite=False, comments=True, authoritative=False)
```

Do basic format agnostic merging.

#### multistring\_to\_rich (mulstring)

Convert a multistring to a list of "rich" string trees:

#### parse (dtdsrc)

read the first dtd element from the source code into this object, return linesprocessed

# removenotes (origin=None)

Remove all the translator's notes.

#### rich source

# See also:

```
rich_to_multistring(), multistring_to_rich()
```

# rich\_target

#### See also:

```
rich_to_multistring(), multistring_to_rich()
```

# classmethod rich\_to\_multistring(elem\_list)

Convert a "rich" string tree to a multistring:

```
>>> from translate.storage.placeables.interfaces import X
>>> rich = [StringElem(['foo', X(id='xxx', sub=[' ']), 'bar'])]
>>> TranslationUnit.rich_to_multistring(rich)
multistring(u'foo bar')
```

```
setcontext(context)
          Set the message context
     setid(new_id)
          Sets the unique identified for this unit.
          only implemented if format allows ids independant from other unit properties like source or context
     source
          gets the unquoted source string
     target
          gets the unquoted target string
     unit iter()
          Iterator that only returns this unit.
translate.storage.dtd.labelsuffixes = ('.label', '.title')
     Label suffixes: entries with this suffix are able to be comibed with accesskeys found in entries ending with
     accesskeysuffixes
translate.storage.dtd.quoteforandroid(source)
     Escapes a line for Android DTD files.
translate.storage.dtd.quotefordtd(source)
     Quotes and escapes a line for regular DTD files.
translate.storage.dtd.removeinvalidamps (name, value)
     Find and remove ampersands that are not part of an entity definition.
     A stray & in a DTD file can break an application's ability to parse the file. In Mozilla localisation this is very
     important and these can break the parsing of files used in XUL and thus break interface rendering. Tracking
     down the problem is very difficult, thus by removing potential broken ampersand and warning the users we can
     ensure that the output DTD will always be parsable.
          Parameters
```

- name (String) Entity name
- value (String) Entity text value

**Return type** String

**Returns** Entity value without bad ampersands

translate.storage.dtd.unquotefromandroid(source)

Unquotes a quoted Android DTD definition.

translate.storage.dtd.unquotefromdtd(source)
unquotes a quoted dtd definition

# \_factory\_classes

Py2exe can't find stuff that we import dynamically, so we have this file just for the sake of the Windows installer to easily pick up all the stuff that we need and ensure they make it into the installer.

### factory

factory methods to build real storage objects that conform to base.py

```
translate.storage.factory.getclass(storefile, localfiletype=None, ignore=None, classes=None,
                                                classes str=None, hiddenclasses=None)
     Factory that returns the applicable class for the type of file presented. Specify ignore to ignore some part at the
     back of the name (like .gz).
translate.storage.factory.getobject(storefile,
                                                                localfiletype=None,
                                                                                         ignore=None,
                                                 classes=None, classes_str=None, hiddenclasses=None)
     Factory that returns a usable object for the type of file presented.
          Parameters storefile (file or str) - File object or file name.
     Specify ignore to ignore some part at the back of the name (like .gz).
translate.storage.factory.supported_files()
     Returns data about all supported files
          Returns list of type that include (name, extensions, mimetypes)
          Return type list
fpo
html
module for parsing html files for translation
class translate.storage.html.POHTMLParser(includeuntaggeddata=None,
                                                                                       inputfile=None,
                                                         callback=None)
     UnitClass
          alias of htmlunit
     add_unit_to_index(unit)
          Add a unit to source and location idexes
     addsourceunit (source)
          Add and returns a new unit with the given source string.
               Return type TranslationUnit
     addunit (unit)
          Append the given unit to the object's list of units.
          This method should always be used rather than trying to modify the list manually.
               Parameters unit (TranslationUnit) – The unit that will be added.
     buildtag (tag, attrs=None, startend=False)
          Create an HTML tag
     close()
          Handle any buffered data.
     detect_encoding (text, default_encodings=None)
          Try to detect a file encoding from text, using either the chardet lib or by trying to decode the file.
     do_encoding(htmlsrc)
          Return the html text properly encoded based on a charset.
     fallback detection(text)
          Simple detection based on BOM in case chardet is not available.
```

```
feed (data)
     Feed data to the parser.
     Call this as often as you want, with as little or as much text as you want (may include 'n').
     find unit with matching id by checking id_index
findunit (source)
     Find the unit with the given source string.
         Return type TranslationUnit or None
findunits (source)
     Find the units with the given source string.
         Return type TranslationUnit or None
get_starttag_text()
     Return full source of start tag: '<...>'.
getids (filename=None)
     return a list of unit ids
getpos()
     Return current line number and offset.
getprojectstyle()
     Get the project type for this store.
getsourcelanguage()
     Get the source language for this store.
gettargetlanguage()
     Get the target language for this store.
getunits()
     Return a list of all units in this store.
guess_encoding(htmlsrc)
     Returns the encoding of the html text.
     We look for 'charset=' within a meta tag to do this.
handle charref(name)
     Handle entries in the form &#NNNN; e.g. ⃡
handle_entityref(name)
     Handle named entities of the form &aaaa; e.g. '
has_translatable_content(text)
     Check if the supplied HTML snippet has any content that needs to be translated.
isempty()
     Return True if the object doesn't contain any translation units.
makeindex()
     Indexes the items in this store. At least .sourceindex should be useful.
merge_on
     The matching criterion to use when merging on.
         Returns The default matching criterion for all the subclasses.
```

4.1. API 433

**Return type** string

#### parse (htmlsrc)

parser to process the given source string

# classmethod parsefile(storefile)

Reads the given file (or opens the given filename) and parses back to an object.

# classmethod parsestring(storestring)

Convert the string representation back to an object.

#### pi escape (text)

Replaces all instances of process instruction with placeholders, and returns the new text and a dictionary of tags. The current implementation replaces <?foo?> with <?md5(foo)?>. The hash => code conversions are stored in self.pidict for later use in restoring the real PHP.

The purpose of this is to remove all potential "tag-like" code from inside PHP. The hash looks nothing like an HTML tag, but the following PHP:

```
$a < $b ? $c : ($d > $e ? $f : $g)
```

looks like it contains an HTML tag:

```
< $b ? $c : ($d >
```

to nearly any regex. Hence, we replace all contents of PHP with simple strings to help our regexes out.

# pi\_unescape(text)

Replaces the PHP placeholders in text with the real code

#### remove\_unit\_from\_index(unit)

Remove a unit from source and locaton indexes

# require\_index()

make sure source index exists

#### reset()

Reset this instance. Loses all unprocessed data.

#### save()

Save to the file that data was originally read from, if available.

# savefile (storefile)

Write the string representation to the given file (or filename).

# serialize(out)

Converts to a bytes representation that can be parsed back using parsestring(). out should be an open file-like objects to write to.

# setprojectstyle (project\_style)

Set the project type for this store.

# setsourcelanguage (sourcelanguage)

Set the source language for this store.

# settargetlanguage (targetlanguage)

Set the target language for this store.

#### translate(source)

Return the translated string for a given source string.

Return type String or None

#### unit iter()

Iterator over all the units in this store.

```
class translate.storage.html.htmlfile(includeuntaggeddata=None, inputfile=None, call-
                                                  back=None)
     INCLUDEATTRS = ['alt', 'abbr', 'content', 'standby', 'summary', 'title']
          Text from these attributes are extracted
     MARKINGATTRS = []
          Text from tags with these attributes will be extracted from the HTML document
     MARKINGTAGS = ['address', 'caption', 'div', 'dt', 'dd', 'figcaption', 'h1', 'h2', 'h3'
          Text in these tags that will be extracted from the HTML document
     SELF_CLOSING_TAGS = [u'area', u'base', u'basefont', u'br', u'col', u'frame', u'hr', u'
          HTML self-closing tags. Tags that should be specified as <img /> but might be <img>. Reference
     UnitClass
          alias of htmlunit
     add unit to index (unit)
          Add a unit to source and location idexes
     addsourceunit (source)
          Add and returns a new unit with the given source string.
              Return type TranslationUnit
     addunit (unit)
          Append the given unit to the object's list of units.
          This method should always be used rather than trying to modify the list manually.
              Parameters unit (TranslationUnit) - The unit that will be added.
     buildtag (tag, attrs=None, startend=False)
          Create an HTML tag
     close()
          Handle any buffered data.
     detect encoding (text, default encodings=None)
          Try to detect a file encoding from text, using either the chardet lib or by trying to decode the file.
     do encoding(htmlsrc)
          Return the html text properly encoded based on a charset.
     fallback_detection (text)
          Simple detection based on BOM in case chardet is not available.
     feed (data)
          Feed data to the parser.
          Call this as often as you want, with as little or as much text as you want (may include 'n').
     findid(id)
          find unit with matching id by checking id_index
     findunit (source)
          Find the unit with the given source string.
              Return type TranslationUnit or None
     findunits(source)
          Find the units with the given source string.
```

4.1. API 435

Return type TranslationUnit or None

```
get_starttag_text()
     Return full source of start tag: '<...>'.
getids (filename=None)
     return a list of unit ids
getpos()
     Return current line number and offset.
getprojectstyle()
     Get the project type for this store.
getsourcelanguage()
     Get the source language for this store.
gettargetlanguage()
     Get the target language for this store.
getunits()
     Return a list of all units in this store.
quess encoding(htmlsrc)
     Returns the encoding of the html text.
     We look for 'charset=' within a meta tag to do this.
handle_charref(name)
     Handle entries in the form &#NNNN; e.g. ⃡
handle_entityref(name)
     Handle named entities of the form &aaaa; e.g. '
has_translatable_content (text)
     Check if the supplied HTML snippet has any content that needs to be translated.
isempty()
     Return True if the object doesn't contain any translation units.
makeindex()
     Indexes the items in this store. At least .sourceindex should be useful.
merge on
     The matching criterion to use when merging on.
         Returns The default matching criterion for all the subclasses.
         Return type string
parse (htmlsrc)
     parser to process the given source string
classmethod parsefile (storefile)
     Reads the given file (or opens the given filename) and parses back to an object.
classmethod parsestring(storestring)
```

Convert the string representation back to an object.

#### pi\_escape (text)

Replaces all instances of process instruction with placeholders, and returns the new text and a dictionary of tags. The current implementation replaces <?foo?> with <?md5(foo)?>. The hash => code conversions are stored in self.pidict for later use in restoring the real PHP.

The purpose of this is to remove all potential "tag-like" code from inside PHP. The hash looks nothing like an HTML tag, but the following PHP:

```
$a < $b ? $c : ($d > $e ? $f : $g)
```

looks like it contains an HTML tag:

```
< $b ? $c : ($d >
```

to nearly any regex. Hence, we replace all contents of PHP with simple strings to help our regexes out.

#### pi\_unescape (text)

Replaces the PHP placeholders in text with the real code

### remove\_unit\_from\_index(unit)

Remove a unit from source and locaton indexes

# require\_index()

make sure source index exists

#### reset()

Reset this instance. Loses all unprocessed data.

#### save()

Save to the file that data was originally read from, if available.

#### savefile (storefile)

Write the string representation to the given file (or filename).

#### serialize (out)

Converts to a bytes representation that can be parsed back using parsestring(). out should be an open file-like objects to write to.

# setprojectstyle (project\_style)

Set the project type for this store.

# $\verb|setsource| language (source| language)|$

Set the source language for this store.

# settargetlanguage (targetlanguage)

Set the target language for this store.

#### translate(source)

Return the translated string for a given source string.

Return type String or None

#### unit\_iter()

Iterator over all the units in this store.

# class translate.storage.html.htmlunit(source=None)

A unit of translatable/localisable HTML content

# adderror (errorname, errortext)

Adds an error message to this unit.

#### **Parameters**

- **errorname** (*string*) A single word to id the error.
- **errortext** (*string*) The text describing the error.

# addlocation (location)

Add one location to the list of locations.

Note: Shouldn't be implemented if the format doesn't support it.

#### addlocations (location)

Add a location or a list of locations.

**Note:** Most classes shouldn't need to implement this, but should rather implement TranslationUnit.addlocation().

Warning: This method might be removed in future.

# addnote (text, origin=None, position='append')

Adds a note (comment).

#### **Parameters**

- **text** (*string*) Usually just a sentence or two.
- **origin** (string) Specifies who/where the comment comes from. Origin can be one of the following text strings: 'translator' 'developer', 'programmer', 'source code' (synonyms)

#### classmethod buildfromunit (unit)

Build a native unit from a foreign unit, preserving as much information as possible.

#### getcontext()

Get the message context.

#### geterrors()

Get all error messages.

### **Return type** Dictionary

# getid()

A unique identifier for this unit.

# **Return type** string

**Returns** an identifier for this unit that is unique in the store

Derived classes should override this in a way that guarantees a unique identifier for each unit in the store.

# getlocations()

A list of source code locations.

#### Return type List

**Note:** Shouldn't be implemented if the format doesn't support it.

# getnotes (origin=None)

Returns all notes about this unit.

It will probably be freeform text or something reasonable that can be synthesised by the format. It should not include location comments (see <code>getlocations()</code>).

#### gettargetlen()

Returns the length of the target string.

#### Return type Integer

Note: Plural forms might be combined.

#### getunits()

This unit in a list.

#### hasplural()

Tells whether or not this specific unit has plural strings.

#### infer\_state()

Empty method that should be overridden in sub-classes to infer the current state(\_n) of the unit from its current state.

#### isblank()

Used to see if this unit has no source or target string.

**Note:** This is probably used more to find translatable units, and we might want to move in that direction rather and get rid of this.

# isfuzzy()

Indicates whether this unit is fuzzy.

#### isheader()

Indicates whether this unit is a header.

# isobsolete()

indicate whether a unit is obsolete

# isreview()

Indicates whether this unit needs review.

#### istranslatable()

Indicates whether this unit can be translated.

This should be used to distinguish real units for translation from header, obsolete, binary or other blank units.

# istranslated()

Indicates whether this unit is translated.

This should be used rather than deducing it from .target, to ensure that other classes can implement more functionality (as XLIFF does).

#### makeobsolete()

Make a unit obsolete

# markfuzzy(value=True)

Marks the unit as fuzzy or not.

# $\verb|markreviewneeded| (needs review=True, explanation=None)$

Marks the unit to indicate whether it needs review.

#### **Parameters**

- needsreview Defaults to True.
- **explanation** Adds an optional explanation as a note.

```
merge (other unit, overwrite = False, comments = True, authoritative = False)
```

Do basic format agnostic merging.

```
multistring_to_rich (mulstring)
```

Convert a multistring to a list of "rich" string trees:

```
>>> target = multistring([u'foo', u'bar', u'baz'])
>>> TranslationUnit.multistring_to_rich(target)
[<StringElem([<StringElem([u'foo'])>])>,
    <StringElem([<StringElem([u'bar'])>])>,
    <StringElem([<StringElem([u'baz'])>])>]
```

# removenotes (origin=None)

Remove all the translator's notes.

#### rich source

#### See also:

```
rich_to_multistring(), multistring_to_rich()
```

#### rich\_target

#### See also:

```
rich_to_multistring(), multistring_to_rich()
```

# classmethod rich\_to\_multistring(elem\_list)

Convert a "rich" string tree to a multistring:

```
>>> from translate.storage.placeables.interfaces import X
>>> rich = [StringElem(['foo', X(id='xxx', sub=[' ']), 'bar'])]
>>> TranslationUnit.rich_to_multistring(rich)
multistring(u'foo bar')
```

# setcontext (context)

Set the message context

# setid(value)

Sets the unique identified for this unit.

only implemented if format allows ids independent from other unit properties like source or context

# unit\_iter()

Iterator that only returns this unit.

```
translate.storage.html.normalize html(text)
```

Remove double spaces from HTML snippets

```
{\tt translate.storage.html.safe\_escape}~({\it html})
```

Escape &, < and >

```
translate.storage.html.strip_html(text)
```

Strip unnecessary html from the text.

HTML tags are deemed unnecessary if it fully encloses the translatable text, eg. '<a href="index.html">Home Page</a>'.

HTML tags that occurs within the normal flow of text will not be removed, eg. 'This is a link to the <a href="index.html">href="index.html">href="index.html">Home Page</a>.'

#### ical

Class that manages iCalender files for translation.

iCalendar files follow the RFC2445 specification.

The iCalendar specification uses the following naming conventions:

- Component: an event, journal entry, timezone, etc
- Property: a property of a component: summary, description, start time, etc
- Attribute: an attribute of a property, e.g. language

The following are localisable in this implementation:

findunits (source)

Find the units with the given source string.

VEVENT component: SUMMARY, DESCRIPTION, COMMENT and LOCATION properties

While other items could be localised this is not seen as important until use cases arise. In such a case simply adjusting the component.name and property.name lists to include these will allow expanded localisation.

**LANGUAGE Attribute** While the iCalendar format allows items to have a language attribute this is not used. The reason being that for most of the items that we localise they are only allowed to occur zero or once. Thus 'summary' would ideally be present in multiple languages in one file, the format does not allow such multiple entries. This is unfortunate as it prevents the creation of a single multilingual iCalendar file.

**Future Format Support** As this format used vobject which supports various formats including vCard it is possible to expand this format to understand those if needed.

```
class translate.storage.ical.icalfile(inputfile=None, **kwargs)
     An ical file
     UnitClass
          alias of icalunit
     add_unit_to_index(unit)
          Add a unit to source and location idexes
     addsourceunit (source)
          Add and returns a new unit with the given source string.
               Return type TranslationUnit
     addunit (unit)
          Append the given unit to the object's list of units.
          This method should always be used rather than trying to modify the list manually.
              Parameters unit (TranslationUnit) - The unit that will be added.
     detect_encoding (text, default_encodings=None)
          Try to detect a file encoding from text, using either the chardet lib or by trying to decode the file.
     fallback detection(text)
          Simple detection based on BOM in case chardet is not available.
     findid(id)
          find unit with matching id by checking id_index
     findunit (source)
          Find the unit with the given source string.
               Return type TranslationUnit or None
```

# Return type TranslationUnit or None getids (filename=None) return a list of unit ids getprojectstyle() Get the project type for this store. getsourcelanguage() Get the source language for this store. gettargetlanguage() Get the target language for this store. getunits() Return a list of all units in this store. isempty() Return True if the object doesn't contain any translation units. makeindex() Indexes the items in this store. At least .sourceindex should be useful. merge on The matching criterion to use when merging on. **Returns** The default matching criterion for all the subclasses. Return type string parse (input) parse the given file or file source string classmethod parsefile (storefile) Reads the given file (or opens the given filename) and parses back to an object. classmethod parsestring(storestring) Convert the string representation back to an object. remove\_unit\_from\_index(unit) Remove a unit from source and locaton indexes require index() make sure source index exists save() Save to the file that data was originally read from, if available. savefile (storefile) Write the string representation to the given file (or filename). serialize (out) Converts to a bytes representation that can be parsed back using parsestring(). out should be an open file-like objects to write to. setprojectstyle (project\_style) Set the project type for this store. setsourcelanguage (sourcelanguage) Set the source language for this store. settargetlanguage (targetlanguage) Set the target language for this store.

#### translate(source)

Return the translated string for a given source string.

# Return type String or None

#### unit iter()

Iterator over all the units in this store.

# class translate.storage.ical.icalunit(source=None, \*\*kwargs)

An ical entry that is translatable

# adderror (errorname, errortext)

Adds an error message to this unit.

#### **Parameters**

- errorname (string) A single word to id the error.
- **errortext** (*string*) The text describing the error.

# addlocation (location)

Add one location to the list of locations.

**Note:** Shouldn't be implemented if the format doesn't support it.

#### addlocations (location)

Add a location or a list of locations.

**Note:** Most classes shouldn't need to implement this, but should rather implement TranslationUnit.addlocation().

**Warning:** This method might be removed in future.

# addnote (text, origin=None, position='append')

Adds a note (comment).

# **Parameters**

- **text** (*string*) Usually just a sentence or two.
- **origin** (string) Specifies who/where the comment comes from. Origin can be one of the following text strings: 'translator' 'developer', 'programmer', 'source code' (synonyms)

# classmethod buildfromunit (unit)

Build a native unit from a foreign unit, preserving as much information as possible.

#### getcontext()

Get the message context.

# geterrors()

Get all error messages.

#### **Return type** Dictionary

#### getid()

A unique identifier for this unit.

Return type string

**Returns** an identifier for this unit that is unique in the store

Derived classes should override this in a way that guarantees a unique identifier for each unit in the store.

# getlocations()

A list of source code locations.

# **Return type** List

Note: Shouldn't be implemented if the format doesn't support it.

#### getnotes (origin=None)

Returns all notes about this unit.

It will probably be freeform text or something reasonable that can be synthesised by the format. It should not include location comments (see getlocations()).

# gettargetlen()

Returns the length of the target string.

# Return type Integer

Note: Plural forms might be combined.

#### getunits()

This unit in a list.

# hasplural()

Tells whether or not this specific unit has plural strings.

#### infer\_state()

Empty method that should be overridden in sub-classes to infer the current state(\_n) of the unit from its current state.

# isblank()

Used to see if this unit has no source or target string.

**Note:** This is probably used more to find translatable units, and we might want to move in that direction rather and get rid of this.

# isfuzzy()

Indicates whether this unit is fuzzy.

#### isheader()

Indicates whether this unit is a header.

# isobsolete()

indicate whether a unit is obsolete

#### isreview()

Indicates whether this unit needs review.

#### istranslatable()

Indicates whether this unit can be translated.

This should be used to distinguish real units for translation from header, obsolete, binary or other blank units.

#### istranslated()

Indicates whether this unit is translated.

This should be used rather than deducing it from .target, to ensure that other classes can implement more functionality (as XLIFF does).

# makeobsolete()

Make a unit obsolete

# markfuzzy(value=True)

Marks the unit as fuzzy or not.

# markreviewneeded (needsreview=True, explanation=None)

Marks the unit to indicate whether it needs review.

#### **Parameters**

- needsreview Defaults to True.
- **explanation** Adds an optional explanation as a note.

```
merge (otherunit, overwrite=False, comments=True, authoritative=False)
```

Do basic format agnostic merging.

# multistring\_to\_rich (mulstring)

Convert a multistring to a list of "rich" string trees:

# removenotes (origin=None)

Remove all the translator's notes.

#### rich\_source

#### See also:

```
rich_to_multistring(), multistring_to_rich()
```

#### rich\_target

# See also:

```
rich_to_multistring(), multistring_to_rich()
```

# classmethod rich\_to\_multistring(elem\_list)

Convert a "rich" string tree to a multistring:

```
>>> from translate.storage.placeables.interfaces import X
>>> rich = [StringElem(['foo', X(id='xxx', sub=[' ']), 'bar'])]
>>> TranslationUnit.rich_to_multistring(rich)
multistring(u'foo bar')
```

# setcontext (context)

Set the message context

#### setid(value)

Sets the unique identified for this unit.

```
only implemented if format allows ids independant from other unit properties like source or context
     unit iter()
          Iterator that only returns this unit.
ini
Class that manages .ini files for translation
# a comment; a comment
[Section] a = a string b : a string
class translate.storage.ini.Dialect
     Base class for differentiating dialect options and functions
class translate.storage.ini.DialectDefault
class translate.storage.ini.DialectInno
class translate.storage.ini.inifile(inputfile=None, dialect='default', **kwargs)
     An INI file
     UnitClass
          alias of iniunit
     add unit to index(unit)
          Add a unit to source and location idexes
     addsourceunit (source)
          Add and returns a new unit with the given source string.
              Return type TranslationUnit
     addunit (unit)
          Append the given unit to the object's list of units.
          This method should always be used rather than trying to modify the list manually.
              Parameters unit (TranslationUnit) - The unit that will be added.
     detect encoding(text, default encodings=None)
          Try to detect a file encoding from text, using either the chardet lib or by trying to decode the file.
     fallback_detection (text)
          Simple detection based on BOM in case chardet is not available.
     findid(id)
          find unit with matching id by checking id_index
     findunit (source)
          Find the unit with the given source string.
              Return type TranslationUnit or None
     findunits (source)
          Find the units with the given source string.
              Return type TranslationUnit or None
     getids (filename=None)
```

return a list of unit ids

# getprojectstyle() Get the project type for this store. getsourcelanguage() Get the source language for this store. gettargetlanguage() Get the target language for this store. getunits() Return a list of all units in this store. isempty() Return True if the object doesn't contain any translation units. makeindex() Indexes the items in this store. At least .sourceindex should be useful. merge\_on The matching criterion to use when merging on. **Returns** The default matching criterion for all the subclasses. **Return type** string parse(input) Parse the given file or file source string. classmethod parsefile (storefile) Reads the given file (or opens the given filename) and parses back to an object. classmethod parsestring(storestring) Convert the string representation back to an object. remove\_unit\_from\_index(unit) Remove a unit from source and locaton indexes require\_index() make sure source index exists save() Save to the file that data was originally read from, if available. savefile (storefile) Write the string representation to the given file (or filename). serialize(out) Converts to a bytes representation that can be parsed back using parsestring(). out should be an open file-like objects to write to. setprojectstyle (project\_style) Set the project type for this store. setsourcelanguage (sourcelanguage) Set the source language for this store.

4.1. API 447

**settargetlanguage** (*targetlanguage*)
Set the target language for this store.

**Return type** String or None

Return the translated string for a given source string.

translate(source)

#### unit iter()

Iterator over all the units in this store.

# class translate.storage.ini.iniunit(source=None, \*\*kwargs)

A INI file entry

#### adderror (errorname, errortext)

Adds an error message to this unit.

#### **Parameters**

- **errorname** (*string*) A single word to id the error.
- **errortext** (*string*) The text describing the error.

#### addlocation (location)

Add one location to the list of locations.

**Note:** Shouldn't be implemented if the format doesn't support it.

#### addlocations (location)

Add a location or a list of locations.

**Note:** Most classes shouldn't need to implement this, but should rather implement TranslationUnit.addlocation().

**Warning:** This method might be removed in future.

# addnote (text, origin=None, position='append')

Adds a note (comment).

#### **Parameters**

- **text** (*string*) Usually just a sentence or two.
- **origin** (*string*) Specifies who/where the comment comes from. Origin can be one of the following text strings: 'translator' 'developer', 'programmer', 'source code' (synonyms)

# classmethod buildfromunit (unit)

Build a native unit from a foreign unit, preserving as much information as possible.

# getcontext()

Get the message context.

# geterrors()

Get all error messages.

# Return type Dictionary

# getid()

A unique identifier for this unit.

#### Return type string

Returns an identifier for this unit that is unique in the store

Derived classes should override this in a way that guarantees a unique identifier for each unit in the store.

#### getlocations()

A list of source code locations.

# Return type List

**Note:** Shouldn't be implemented if the format doesn't support it.

# getnotes (origin=None)

Returns all notes about this unit.

It will probably be freeform text or something reasonable that can be synthesised by the format. It should not include location comments (see getlocations()).

#### gettargetlen()

Returns the length of the target string.

# Return type Integer

**Note:** Plural forms might be combined.

# getunits()

This unit in a list.

#### hasplural()

Tells whether or not this specific unit has plural strings.

# infer\_state()

Empty method that should be overridden in sub-classes to infer the current state(\_n) of the unit from its current state.

### isblank()

Used to see if this unit has no source or target string.

**Note:** This is probably used more to find translatable units, and we might want to move in that direction rather and get rid of this.

# isfuzzy()

Indicates whether this unit is fuzzy.

#### isheader()

Indicates whether this unit is a header.

#### isobsolete()

indicate whether a unit is obsolete

# isreview()

Indicates whether this unit needs review.

#### istranslatable()

Indicates whether this unit can be translated.

This should be used to distinguish real units for translation from header, obsolete, binary or other blank units.

# istranslated()

Indicates whether this unit is translated.

This should be used rather than deducing it from .target, to ensure that other classes can implement more functionality (as XLIFF does).

# makeobsolete()

Make a unit obsolete

# markfuzzy (value=True)

Marks the unit as fuzzy or not.

#### markreviewneeded (needsreview=True, explanation=None)

Marks the unit to indicate whether it needs review.

#### **Parameters**

- needsreview Defaults to True.
- **explanation** Adds an optional explanation as a note.

```
merge (otherunit, overwrite=False, comments=True, authoritative=False)
```

Do basic format agnostic merging.

#### multistring\_to\_rich(mulstring)

Convert a multistring to a list of "rich" string trees:

#### removenotes (origin=None)

Remove all the translator's notes.

#### rich source

#### See also:

```
rich_to_multistring(), multistring_to_rich()
```

# rich\_target

#### See also:

```
rich_to_multistring(), multistring_to_rich()
```

# classmethod rich\_to\_multistring(elem\_list)

Convert a "rich" string tree to a multistring:

```
>>> from translate.storage.placeables.interfaces import X
>>> rich = [StringElem(['foo', X(id='xxx', sub=[' ']), 'bar'])]
>>> TranslationUnit.rich_to_multistring(rich)
multistring(u'foo bar')
```

# setcontext (context)

Set the message context

# setid(value)

Sets the unique identified for this unit.

only implemented if format allows ids independant from other unit properties like source or context

```
unit iter()
```

Iterator that only returns this unit.

```
translate.storage.ini.register_dialect (dialect)
```

Decorator that registers the dialect.

# jsonl10n

Class that manages JSON data files for translation

JSON is an acronym for JavaScript Object Notation, it is an open standard designed for human-readable data interchange.

JSON basic types:

- Number (integer or real)
- String (double-quoted Unicode with backslash escaping)
- Boolean (true or false)
- Array (an ordered sequence of values, comma-separated and enclosed in square brackets)
- Object (a collection of key:value pairs, comma-separated and enclosed in curly braces)
- null

# Example:

# TODO:

- Handle \u and other escapes in Unicode
- Manage data type storage and conversion. True -> "True" -> True

**class** translate.storage.jsonl10n.**I18NextFile**(*inputfile=None*, *filter=None*, \*\*kwargs)
A i18next v3 format, this is nested JSON with several additions.

See https://www.i18next.com/

# UnitClass alias of I18NextUnit add\_unit\_to\_index(unit) Add a unit to source and location idexes addsourceunit (source) Add and returns a new unit with the given source string. Return type TranslationUnit addunit (unit) Append the given unit to the object's list of units. This method should always be used rather than trying to modify the list manually. **Parameters unit** (TranslationUnit) – The unit that will be added. detect\_encoding(text, default\_encodings=None) Try to detect a file encoding from text, using either the chardet lib or by trying to decode the file. fallback detection(text) Simple detection based on BOM in case chardet is not available. findid(id) find unit with matching id by checking id\_index findunit (source) Find the unit with the given source string. **Return type** TranslationUnit or None findunits (source) Find the units with the given source string. Return type TranslationUnit or None getids (filename=None) return a list of unit ids getprojectstyle() Get the project type for this store. getsourcelanguage() Get the source language for this store. gettargetlanguage() Get the target language for this store.

# getunits()

Return a list of all units in this store.

#### Retain

isempty()

# Return True if the object doesn't contain any translation units.

makeindex ()

Indexes the items in this store. At least .sourceindex should be useful.

#### merge on

The matching criterion to use when merging on.

**Returns** The default matching criterion for all the subclasses.

Return type string

```
parse (input)
          parse the given file or file source string
     classmethod parsefile(storefile)
          Reads the given file (or opens the given filename) and parses back to an object.
     classmethod parsestring(storestring)
          Convert the string representation back to an object.
     remove unit from index(unit)
          Remove a unit from source and locaton indexes
     require_index()
          make sure source index exists
     save()
          Save to the file that data was originally read from, if available.
     savefile (storefile)
          Write the string representation to the given file (or filename).
     serialize(out)
          Converts to a bytes representation that can be parsed back using parsestring(). out should be an
          open file-like objects to write to.
     setprojectstyle (project_style)
          Set the project type for this store.
     setsourcelanguage (sourcelanguage)
          Set the source language for this store.
     settargetlanguage (targetlanguage)
          Set the target language for this store.
     translate(source)
          Return the translated string for a given source string.
               Return type String or None
     unit_iter()
          Iterator over all the units in this store.
class translate.storage.jsonl10n.I18NextUnit (source=None, item=None, notes=None,
                                                              placeholders=None, **kwargs)
     A i18next v3 format, JSON with plurals.
     See https://www.i18next.com/
     adderror (errorname, errortext)
          Adds an error message to this unit.
               Parameters
```

- **errorname** (*string*) A single word to id the error.
- **errortext** (*string*) The text describing the error.

# addlocation (location)

Add one location to the list of locations.

**Note:** Shouldn't be implemented if the format doesn't support it.

#### addlocations (location)

Add a location or a list of locations.

**Note:** Most classes shouldn't need to implement this, but should rather implement TranslationUnit.addlocation().

Warning: This method might be removed in future.

# addnote (text, origin=None, position='append')

Adds a note (comment).

#### **Parameters**

- **text** (*string*) Usually just a sentence or two.
- **origin** (*string*) Specifies who/where the comment comes from. Origin can be one of the following text strings: 'translator' 'developer', 'programmer', 'source code' (synonyms)

#### classmethod buildfromunit (unit)

Build a native unit from a foreign unit, preserving as much information as possible.

#### getcontext()

Get the message context.

#### geterrors()

Get all error messages.

#### **Return type** Dictionary

#### getid()

A unique identifier for this unit.

# **Return type** string

**Returns** an identifier for this unit that is unique in the store

Derived classes should override this in a way that guarantees a unique identifier for each unit in the store.

#### getlocations()

A list of source code locations.

# **Return type** List

**Note:** Shouldn't be implemented if the format doesn't support it.

# getnotes (origin=None)

Returns all notes about this unit.

It will probably be freeform text or something reasonable that can be synthesised by the format. It should not include location comments (see <code>getlocations()</code>).

# gettargetlen()

Returns the length of the target string.

# Return type Integer

Note: Plural forms might be combined.

# getunits()

This unit in a list.

# getvalue()

Return value to be stored in JSON file.

# hasplural()

Tells whether or not this specific unit has plural strings.

#### infer\_state()

Empty method that should be overridden in sub-classes to infer the current state(\_n) of the unit from its current state.

# isblank()

Used to see if this unit has no source or target string.

**Note:** This is probably used more to find translatable units, and we might want to move in that direction rather and get rid of this.

#### isfuzzy()

Indicates whether this unit is fuzzy.

#### isheader()

Indicates whether this unit is a header.

#### isobsolete()

indicate whether a unit is obsolete

# isreview()

Indicates whether this unit needs review.

# istranslatable()

Indicates whether this unit can be translated.

This should be used to distinguish real units for translation from header, obsolete, binary or other blank units.

#### istranslated()

Indicates whether this unit is translated.

This should be used rather than deducing it from .target, to ensure that other classes can implement more functionality (as XLIFF does).

#### makeobsolete()

Make a unit obsolete

# markfuzzy(value=True)

Marks the unit as fuzzy or not.

# markreviewneeded (needsreview=True, explanation=None)

Marks the unit to indicate whether it needs review.

# **Parameters**

- needsreview Defaults to True.
- **explanation** Adds an optional explanation as a note.

```
merge (otherunit, overwrite=False, comments=True, authoritative=False)
         Do basic format agnostic merging.
     multistring_to_rich (mulstring)
         Convert a multistring to a list of "rich" string trees:
         >>> target = multistring([u'foo', u'bar', u'baz'])
         >>> TranslationUnit.multistring_to_rich(target)
         [<StringElem([<StringElem([u'foo'])>])>,
          <StringElem([<StringElem([u'bar'])>])>,
          <StringElem([<StringElem([u'baz'])>])>]
     removenotes (origin=None)
         Remove all the translator's notes.
     rich source
         See also:
         rich_to_multistring(), multistring_to_rich()
     rich_target
         See also:
         rich_to_multistring(), multistring_to_rich()
     classmethod rich_to_multistring(elem_list)
         Convert a "rich" string tree to a multistring:
         >>> from translate.storage.placeables.interfaces import X
         >>> rich = [StringElem(['foo', X(id='xxx', sub=[' ']), 'bar'])]
         >>> TranslationUnit.rich_to_multistring(rich)
         multistring(u'foo bar')
     setcontext (context)
         Set the message context
     setid(value)
         Sets the unique identified for this unit.
         only implemented if format allows ids independant from other unit properties like source or context
     unit_iter()
         Iterator that only returns this unit.
class translate.storage.jsonl10n.JsonFile(inputfile=None, filter=None, **kwargs)
     A JSON file
     UnitClass
         alias of JsonUnit
     add_unit_to_index(unit)
         Add a unit to source and location idexes
     addsourceunit (source)
         Add and returns a new unit with the given source string.
             Return type TranslationUnit
```

#### addunit (unit)

Append the given unit to the object's list of units.

This method should always be used rather than trying to modify the list manually.

Parameters unit (TranslationUnit) - The unit that will be added.

# detect\_encoding (text, default\_encodings=None)

Try to detect a file encoding from *text*, using either the chardet lib or by trying to decode the file.

# fallback\_detection (text)

Simple detection based on BOM in case chardet is not available.

# findid(id)

find unit with matching id by checking id\_index

# findunit (source)

Find the unit with the given source string.

Return type TranslationUnit or None

#### findunits (source)

Find the units with the given source string.

Return type TranslationUnit or None

#### getids (filename=None)

return a list of unit ids

#### getprojectstyle()

Get the project type for this store.

# getsourcelanguage()

Get the source language for this store.

# gettargetlanguage()

Get the target language for this store.

# getunits()

Return a list of all units in this store.

#### isempty()

Return True if the object doesn't contain any translation units.

# makeindex()

Indexes the items in this store. At least .sourceindex should be useful.

# merge\_on

The matching criterion to use when merging on.

**Returns** The default matching criterion for all the subclasses.

Return type string

# parse(input)

parse the given file or file source string

#### classmethod parsefile (storefile)

Reads the given file (or opens the given filename) and parses back to an object.

#### classmethod parsestring(storestring)

Convert the string representation back to an object.

# remove\_unit\_from\_index(unit)

Remove a unit from source and locaton indexes

```
require index()
           make sure source index exists
     save()
           Save to the file that data was originally read from, if available.
     savefile (storefile)
           Write the string representation to the given file (or filename).
     serialize (out)
           Converts to a bytes representation that can be parsed back using parsestring (). out should be an
           open file-like objects to write to.
     setprojectstyle (project_style)
           Set the project type for this store.
     setsourcelanguage (sourcelanguage)
           Set the source language for this store.
     settargetlanguage (targetlanguage)
           Set the target language for this store.
     translate(source)
           Return the translated string for a given source string.
               Return type String or None
     unit iter()
          Iterator over all the units in this store.
class translate.storage.jsonl10n.JsonNestedFile(inputfile=None,
                                                                                           filter=None,
                                                                  **kwargs)
     A JSON file with nested keys
     UnitClass
           alias of JsonNestedUnit
     add_unit_to_index(unit)
           Add a unit to source and location idexes
     addsourceunit (source)
           Add and returns a new unit with the given source string.
               Return type TranslationUnit
     addunit (unit)
           Append the given unit to the object's list of units.
           This method should always be used rather than trying to modify the list manually.
               Parameters unit (TranslationUnit) - The unit that will be added.
     detect encoding (text, default encodings=None)
           Try to detect a file encoding from text, using either the chardet lib or by trying to decode the file.
     fallback_detection (text)
           Simple detection based on BOM in case chardet is not available.
     findid(id)
           find unit with matching id by checking id_index
     findunit (source)
           Find the unit with the given source string.
               Return type TranslationUnit or None
```

# Find the units with the given source string. Return type TranslationUnit or None getids (filename=None) return a list of unit ids getprojectstyle() Get the project type for this store. getsourcelanguage() Get the source language for this store. gettargetlanguage() Get the target language for this store. getunits() Return a list of all units in this store. isempty() Return True if the object doesn't contain any translation units. makeindex() Indexes the items in this store. At least .sourceindex should be useful. merge\_on The matching criterion to use when merging on. **Returns** The default matching criterion for all the subclasses. Return type string parse(input) parse the given file or file source string classmethod parsefile (storefile) Reads the given file (or opens the given filename) and parses back to an object. classmethod parsestring(storestring) Convert the string representation back to an object. remove\_unit\_from\_index(unit) Remove a unit from source and locaton indexes require\_index() make sure source index exists save() Save to the file that data was originally read from, if available. savefile (storefile) Write the string representation to the given file (or filename). serialize (out) Converts to a bytes representation that can be parsed back using parsestring (). out should be an open file-like objects to write to. setprojectstyle (project\_style) Set the project type for this store.

findunits (source)

4.1. API 459

setsourcelanguage (sourcelanguage)
Set the source language for this store.

# settargetlanguage (targetlanguage)

Set the target language for this store.

#### translate(source)

Return the translated string for a given source string.

**Return type** String or None

# unit\_iter()

Iterator over all the units in this store.

# adderror (errorname, errortext)

Adds an error message to this unit.

#### **Parameters**

- errorname (string) A single word to id the error.
- **errortext** (*string*) The text describing the error.

#### addlocation (location)

Add one location to the list of locations.

**Note:** Shouldn't be implemented if the format doesn't support it.

#### addlocations (location)

Add a location or a list of locations.

**Note:** Most classes shouldn't need to implement this, but should rather implement TranslationUnit.addlocation().

**Warning:** This method might be removed in future.

addnote (text, origin=None, position='append')

Adds a note (comment).

#### **Parameters**

- **text** (*string*) Usually just a sentence or two.
- **origin** (*string*) Specifies who/where the comment comes from. Origin can be one of the following text strings: 'translator' 'developer', 'programmer', 'source code' (synonyms)

# classmethod buildfromunit (unit)

Build a native unit from a foreign unit, preserving as much information as possible.

# getcontext()

Get the message context.

#### geterrors()

Get all error messages.

Return type Dictionary

#### getid()

A unique identifier for this unit.

# Return type string

Returns an identifier for this unit that is unique in the store

Derived classes should override this in a way that guarantees a unique identifier for each unit in the store.

#### getlocations()

A list of source code locations.

# **Return type** List

Note: Shouldn't be implemented if the format doesn't support it.

# getnotes (origin=None)

Returns all notes about this unit.

It will probably be freeform text or something reasonable that can be synthesised by the format. It should not include location comments (see getlocations()).

# gettargetlen()

Returns the length of the target string.

#### **Return type** Integer

Note: Plural forms might be combined.

# getunits()

This unit in a list.

#### getvalue()

Return value to be stored in JSON file.

#### hasplural()

Tells whether or not this specific unit has plural strings.

#### infer\_state()

Empty method that should be overridden in sub-classes to infer the current state(\_n) of the unit from its current state

#### isblank()

Used to see if this unit has no source or target string.

**Note:** This is probably used more to find translatable units, and we might want to move in that direction rather and get rid of this.

# isfuzzy()

Indicates whether this unit is fuzzy.

#### isheader()

Indicates whether this unit is a header.

# isobsolete()

indicate whether a unit is obsolete

#### isreview()

Indicates whether this unit needs review.

#### istranslatable()

Indicates whether this unit can be translated.

This should be used to distinguish real units for translation from header, obsolete, binary or other blank units.

#### istranslated()

Indicates whether this unit is translated.

This should be used rather than deducing it from .target, to ensure that other classes can implement more functionality (as XLIFF does).

#### makeobsolete()

Make a unit obsolete

#### markfuzzy (value=True)

Marks the unit as fuzzy or not.

#### markreviewneeded (needsreview=True, explanation=None)

Marks the unit to indicate whether it needs review.

#### **Parameters**

- needsreview Defaults to True.
- **explanation** Adds an optional explanation as a note.

**merge** (otherunit, overwrite=False, comments=True, authoritative=False)

Do basic format agnostic merging.

# multistring\_to\_rich(mulstring)

Convert a multistring to a list of "rich" string trees:

#### removenotes (origin=None)

Remove all the translator's notes.

#### rich\_source

# See also:

```
rich_to_multistring(), multistring_to_rich()
```

# rich\_target

# See also:

```
rich_to_multistring(), multistring_to_rich()
```

# ${\tt classmethod\ rich\_to\_multistring}\,(\textit{elem\_list})$

Convert a "rich" string tree to a multistring:

```
>>> from translate.storage.placeables.interfaces import X
>>> rich = [StringElem(['foo', X(id='xxx', sub=[' ']), 'bar'])]
>>> TranslationUnit.rich_to_multistring(rich)
multistring(u'foo bar')
```

#### setcontext (context)

Set the message context

#### setid(value)

Sets the unique identified for this unit.

only implemented if format allows ids independant from other unit properties like source or context

# unit\_iter()

Iterator that only returns this unit.

A JSON entry

# adderror (errorname, errortext)

Adds an error message to this unit.

#### **Parameters**

- **errorname** (*string*) A single word to id the error.
- **errortext** (*string*) The text describing the error.

#### addlocation (location)

Add one location to the list of locations.

**Note:** Shouldn't be implemented if the format doesn't support it.

### addlocations (location)

Add a location or a list of locations.

**Note:** Most classes shouldn't need to implement this, but should rather implement TranslationUnit.addlocation().

**Warning:** This method might be removed in future.

addnote (text, origin=None, position='append')

Adds a note (comment).

#### **Parameters**

- **text** (*string*) Usually just a sentence or two.
- **origin** (*string*) Specifies who/where the comment comes from. Origin can be one of the following text strings: 'translator' 'developer', 'programmer', 'source code' (synonyms)

# classmethod buildfromunit (unit)

Build a native unit from a foreign unit, preserving as much information as possible.

#### getcontext()

Get the message context.

# geterrors()

Get all error messages.

# **Return type** Dictionary

# getid()

A unique identifier for this unit.

# Return type string

Returns an identifier for this unit that is unique in the store

Derived classes should override this in a way that guarantees a unique identifier for each unit in the store.

#### getlocations()

A list of source code locations.

# Return type List

**Note:** Shouldn't be implemented if the format doesn't support it.

# getnotes (origin=None)

Returns all notes about this unit.

It will probably be freeform text or something reasonable that can be synthesised by the format. It should not include location comments (see getlocations()).

# gettargetlen()

Returns the length of the target string.

# Return type Integer

Note: Plural forms might be combined.

#### getunits()

This unit in a list.

# getvalue()

Return value to be stored in JSON file.

# hasplural()

Tells whether or not this specific unit has plural strings.

#### infer\_state()

Empty method that should be overridden in sub-classes to infer the current state(\_n) of the unit from its current state.

#### isblank()

Used to see if this unit has no source or target string.

**Note:** This is probably used more to find translatable units, and we might want to move in that direction rather and get rid of this.

#### isfuzzy()

Indicates whether this unit is fuzzy.

#### isheader()

Indicates whether this unit is a header.

### isobsolete()

indicate whether a unit is obsolete

## isreview()

Indicates whether this unit needs review.

#### istranslatable()

Indicates whether this unit can be translated.

This should be used to distinguish real units for translation from header, obsolete, binary or other blank units.

### istranslated()

Indicates whether this unit is translated.

This should be used rather than deducing it from .target, to ensure that other classes can implement more functionality (as XLIFF does).

### makeobsolete()

Make a unit obsolete

# markfuzzy(value=True)

Marks the unit as fuzzy or not.

## markreviewneeded (needsreview=True, explanation=None)

Marks the unit to indicate whether it needs review.

#### **Parameters**

- needsreview Defaults to True.
- **explanation** Adds an optional explanation as a note.

merge (otherunit, overwrite=False, comments=True, authoritative=False)

Do basic format agnostic merging.

### multistring\_to\_rich(mulstring)

Convert a multistring to a list of "rich" string trees:

# removenotes (origin=None)

Remove all the translator's notes.

### rich source

# See also:

```
rich_to_multistring(), multistring_to_rich()
```

# rich target

### See also:

```
rich_to_multistring(), multistring_to_rich()
```

### classmethod rich\_to\_multistring(elem\_list)

Convert a "rich" string tree to a multistring:

```
>>> from translate.storage.placeables.interfaces import X
>>> rich = [StringElem(['foo', X(id='xxx', sub=[' ']), 'bar'])]
>>> TranslationUnit.rich_to_multistring(rich)
multistring(u'foo bar')
```

#### setcontext (context)

Set the message context

### setid(value)

Sets the unique identified for this unit.

only implemented if format allows ids independant from other unit properties like source or context

### unit iter()

Iterator that only returns this unit.

WebExtension JSON file

See following URLs for doc:

https://developer.chrome.com/extensions/i18n https://developer.mozilla.org/en-US/Add-ons/WebExtensions/Internationalization

### UnitClass

alias of WebExtensionJsonUnit

#### add unit to index (unit)

Add a unit to source and location idexes

# addsourceunit (source)

Add and returns a new unit with the given source string.

```
Return type TranslationUnit
```

```
addunit (unit)
```

Append the given unit to the object's list of units.

This method should always be used rather than trying to modify the list manually.

Parameters unit (TranslationUnit) - The unit that will be added.

```
detect_encoding(text, default_encodings=None)
```

Try to detect a file encoding from *text*, using either the chardet lib or by trying to decode the file.

# ${\tt fallback\_detection}\ (\textit{text})$

Simple detection based on BOM in case chardet is not available.

## findid(id)

find unit with matching id by checking id\_index

# findunit (source)

Find the unit with the given source string.

Return type TranslationUnit or None

# findunits (source)

Find the units with the given source string.

Return type TranslationUnit or None

```
getids (filename=None)
     return a list of unit ids
getprojectstyle()
     Get the project type for this store.
getsourcelanguage()
     Get the source language for this store.
gettargetlanguage()
     Get the target language for this store.
getunits()
     Return a list of all units in this store.
isempty()
     Return True if the object doesn't contain any translation units.
makeindex()
     Indexes the items in this store. At least .sourceindex should be useful.
merge on
     The matching criterion to use when merging on.
         Returns The default matching criterion for all the subclasses.
         Return type string
parse (input)
     parse the given file or file source string
classmethod parsefile(storefile)
     Reads the given file (or opens the given filename) and parses back to an object.
classmethod parsestring(storestring)
     Convert the string representation back to an object.
remove_unit_from_index(unit)
     Remove a unit from source and locaton indexes
require_index()
     make sure source index exists
save()
     Save to the file that data was originally read from, if available.
savefile (storefile)
     Write the string representation to the given file (or filename).
serialize (out)
     Converts to a bytes representation that can be parsed back using parsestring(). out should be an
     open file-like objects to write to.
setprojectstyle (project_style)
     Set the project type for this store.
setsourcelanguage (sourcelanguage)
     Set the source language for this store.
settargetlanguage (targetlanguage)
     Set the target language for this store.
```

4.1. API 467

translate(source)

Return the translated string for a given source string.

### **Return type** String or None

### unit iter()

Iterator over all the units in this store.

### adderror (errorname, errortext)

Adds an error message to this unit.

### **Parameters**

- **errorname** (*string*) A single word to id the error.
- **errortext** (*string*) The text describing the error.

### addlocation (location)

Add one location to the list of locations.

**Note:** Shouldn't be implemented if the format doesn't support it.

### addlocations (location)

Add a location or a list of locations.

**Note:** Most classes shouldn't need to implement this, but should rather implement TranslationUnit.addlocation().

**Warning:** This method might be removed in future.

# addnote (text, origin=None, position='append')

Adds a note (comment).

#### **Parameters**

- **text** (*string*) Usually just a sentence or two.
- **origin** (string) Specifies who/where the comment comes from. Origin can be one of the following text strings: 'translator' 'developer', 'programmer', 'source code' (synonyms)

# classmethod buildfromunit (unit)

Build a native unit from a foreign unit, preserving as much information as possible.

# getcontext()

Get the message context.

### geterrors()

Get all error messages.

# Return type Dictionary

### getid()

A unique identifier for this unit.

**Return type** string

**Returns** an identifier for this unit that is unique in the store

Derived classes should override this in a way that guarantees a unique identifier for each unit in the store.

# getlocations()

A list of source code locations.

### **Return type** List

Note: Shouldn't be implemented if the format doesn't support it.

### getnotes (origin=None)

Returns all notes about this unit.

It will probably be freeform text or something reasonable that can be synthesised by the format. It should not include location comments (see getlocations()).

# gettargetlen()

Returns the length of the target string.

# Return type Integer

Note: Plural forms might be combined.

### getunits()

This unit in a list.

# getvalue()

Return value to be stored in JSON file.

# hasplural()

Tells whether or not this specific unit has plural strings.

## infer\_state()

Empty method that should be overridden in sub-classes to infer the current state(\_n) of the unit from its current state.

### isblank()

Used to see if this unit has no source or target string.

**Note:** This is probably used more to find translatable units, and we might want to move in that direction rather and get rid of this.

### isfuzzy()

Indicates whether this unit is fuzzy.

# isheader()

Indicates whether this unit is a header.

# $\verb|isobsolete|()|$

indicate whether a unit is obsolete

# isreview()

Indicates whether this unit needs review.

# istranslatable()

Indicates whether this unit can be translated.

This should be used to distinguish real units for translation from header, obsolete, binary or other blank units.

# istranslated()

Indicates whether this unit is translated.

This should be used rather than deducing it from .target, to ensure that other classes can implement more functionality (as XLIFF does).

#### makeobsolete()

Make a unit obsolete

# markfuzzy(value=True)

Marks the unit as fuzzy or not.

# markreviewneeded (needsreview=True, explanation=None)

Marks the unit to indicate whether it needs review.

### **Parameters**

- needsreview Defaults to True.
- **explanation** Adds an optional explanation as a note.

```
merge (otherunit, overwrite=False, comments=True, authoritative=False)
```

Do basic format agnostic merging.

# multistring\_to\_rich (mulstring)

Convert a multistring to a list of "rich" string trees:

#### removenotes (origin=None)

Remove all the translator's notes.

# rich\_source

### See also:

```
rich_to_multistring(), multistring_to_rich()
```

### rich\_target

### See also:

```
rich to multistring(), multistring to rich()
```

# classmethod rich\_to\_multistring(elem\_list)

Convert a "rich" string tree to a multistring:

```
>>> from translate.storage.placeables.interfaces import X
>>> rich = [StringElem(['foo', X(id='xxx', sub=[' ']), 'bar'])]
>>> TranslationUnit.rich_to_multistring(rich)
multistring(u'foo bar')
```

### setcontext (context)

Set the message context

```
setid(value)
           Sets the unique identified for this unit.
           only implemented if format allows ids independant from other unit properties like source or context
     unit iter()
          Iterator that only returns this unit.
lisa
Parent class for LISA standards (TMX, TBX, XLIFF)
class translate.storage.lisa.LISAfile (inputfile=None,
                                                                        sourcelanguage='en',
                                                                                                 target-
                                                     language=None, **kwargs)
     A class representing a file store for one of the LISA file formats.
     UnitClass
           alias of LISAunit
     add_unit_to_index(unit)
           Add a unit to source and location idexes
     addheader()
           Method to be overridden to initialise headers, etc.
     addsourceunit (source)
           Adds and returns a new unit with the given string as first entry.
     addunit (unit, new=True)
           Append the given unit to the object's list of units.
           This method should always be used rather than trying to modify the list manually.
               Parameters unit (TranslationUnit) - The unit that will be added.
     detect_encoding (text, default_encodings=None)
           Try to detect a file encoding from text, using either the chardet lib or by trying to decode the file.
     fallback_detection (text)
           Simple detection based on BOM in case chardet is not available.
           find unit with matching id by checking id_index
     findunit (source)
           Find the unit with the given source string.
               Return type TranslationUnit or None
     findunits (source)
           Find the units with the given source string.
               Return type TranslationUnit or None
     getids (filename=None)
           return a list of unit ids
     getprojectstyle()
           Get the project type for this store.
     getsourcelanguage()
           Get the source language for this store.
```

### gettargetlanguage()

Get the target language for this store.

### getunits()

Return a list of all units in this store.

### initbody()

Initialises self.body so it never needs to be retrieved from the XML again.

#### isempty()

Return True if the object doesn't contain any translation units.

#### makeindex()

Indexes the items in this store. At least .sourceindex should be useful.

### merge\_on

The matching criterion to use when merging on.

**Returns** The default matching criterion for all the subclasses.

Return type string

### namespaced(name)

Returns name in Clark notation.

For example namespaced ("source") in an XLIFF document might return:

```
{urn:oasis:names:tc:xliff:document:1.1}source
```

This is needed throughout lxml.

### parse (xml)

Populates this object from the given xml string

# classmethod parsefile(storefile)

Reads the given file (or opens the given filename) and parses back to an object.

# classmethod parsestring(storestring)

Convert the string representation back to an object.

### remove\_unit\_from\_index(unit)

Remove a unit from source and locaton indexes

# require\_index()

make sure source index exists

#### save()

Save to the file that data was originally read from, if available.

### savefile (storefile)

Write the string representation to the given file (or filename).

# serialize(out=None)

Converts to a string containing the file's XML

# setprojectstyle (project\_style)

Set the project type for this store.

# setsourcelanguage (sourcelanguage)

Set the source language for this store.

# settargetlanguage (targetlanguage)

Set the target language for this store.

#### translate(source)

Return the translated string for a given source string.

# Return type String or None

## unit\_iter()

Iterator over all the units in this store.

# class translate.storage.lisa.LISAunit (source, empty=False, \*\*kwargs)

A single unit in the file. Provisional work is done to make several languages possible.

### adderror (errorname, errortext)

Adds an error message to this unit.

#### **Parameters**

- errorname (string) A single word to id the error.
- **errortext** (*string*) The text describing the error.

# addlocation (location)

Add one location to the list of locations.

**Note:** Shouldn't be implemented if the format doesn't support it.

### addlocations (location)

Add a location or a list of locations.

**Note:** Most classes shouldn't need to implement this, but should rather implement TranslationUnit.addlocation().

**Warning:** This method might be removed in future.

# addnote (text, origin=None, position='append')

Adds a note (comment).

### **Parameters**

- **text** (*string*) Usually just a sentence or two.
- **origin** (*string*) Specifies who/where the comment comes from. Origin can be one of the following text strings: 'translator' 'developer', 'programmer', 'source code' (synonyms)

# classmethod buildfromunit (unit)

Build a native unit from a foreign unit, preserving as much information as possible.

### createlanguageNode (lang, text, purpose=None)

Returns a xml Element setup with given parameters to represent a single language entry. Has to be over-ridden.

# getNodeText (languageNode, xml\_space='preserve')

Retrieves the term from the given languageNode.

### getcontext()

Get the message context.

#### geterrors()

Get all error messages.

# **Return type** Dictionary

### getid()

A unique identifier for this unit.

# Return type string

**Returns** an identifier for this unit that is unique in the store

Derived classes should override this in a way that guarantees a unique identifier for each unit in the store.

### getlanguageNode (lang=None, index=None)

Retrieves a languageNode either by language or by index.

# getlanguageNodes()

Returns a list of all nodes that contain per language information.

### getlocations()

A list of source code locations.

### **Return type** List

**Note:** Shouldn't be implemented if the format doesn't support it.

### getnotes (origin=None)

Returns all notes about this unit.

It will probably be freeform text or something reasonable that can be synthesised by the format. It should not include location comments (see <code>getlocations()</code>).

# gettarget (lang=None)

retrieves the "target" text (second entry), or the entry in the specified language, if it exists

# gettargetlen()

Returns the length of the target string.

### **Return type** Integer

**Note:** Plural forms might be combined.

# getunits()

This unit in a list.

### hasplural()

Tells whether or not this specific unit has plural strings.

# infer\_state()

Empty method that should be overridden in sub-classes to infer the current state(\_n) of the unit from its current state.

#### isblank()

Used to see if this unit has no source or target string.

**Note:** This is probably used more to find translatable units, and we might want to move in that direction rather and get rid of this.

### isfuzzy()

Indicates whether this unit is fuzzy.

#### isheader()

Indicates whether this unit is a header.

### isobsolete()

indicate whether a unit is obsolete

#### isreview()

Indicates whether this unit needs review.

### istranslatable()

Indicates whether this unit can be translated.

This should be used to distinguish real units for translation from header, obsolete, binary or other blank units.

## istranslated()

Indicates whether this unit is translated.

This should be used rather than deducing it from .target, to ensure that other classes can implement more functionality (as XLIFF does).

#### makeobsolete()

Make a unit obsolete

### markfuzzy (value=True)

Marks the unit as fuzzy or not.

### markreviewneeded (needsreview=True, explanation=None)

Marks the unit to indicate whether it needs review.

### **Parameters**

- needsreview Defaults to True.
- **explanation** Adds an optional explanation as a note.

**merge** (otherunit, overwrite=False, comments=True, authoritative=False)

Do basic format agnostic merging.

# multistring\_to\_rich (mulstring)

Convert a multistring to a list of "rich" string trees:

### namespaced(name)

Returns name in Clark notation.

For example namespaced ("source") in an XLIFF document might return:

```
{urn:oasis:names:tc:xliff:document:1.1}source
```

This is needed throughout lxml.

### removenotes (origin=None)

Remove all the translator's notes.

```
rich source
    See also:
    rich_to_multistring(), multistring_to_rich()
rich target
    See also:
    rich_to_multistring(), multistring_to_rich()
classmethod rich_to_multistring(elem_list)
    Convert a "rich" string tree to a multistring:
    >>> from translate.storage.placeables.interfaces import X
    >>> rich = [StringElem(['foo', X(id='xxx', sub=[' ']), 'bar'])]
    >>> TranslationUnit.rich_to_multistring(rich)
    multistring(u'foo bar')
setcontext(context)
    Set the message context
setid(value)
    Sets the unique identified for this unit.
    only implemented if format allows ids independant from other unit properties like source or context
settarget (target, lang='xx', append=False)
    Sets the "target" string (second language), or alternatively appends to the list
unit_iter()
    Iterator that only returns this unit.
```

# mo

Module for parsing Gettext .mo files for translation.

The coding of .mo files was produced from Gettext documentation, Pythons msgfmt.py and by observing and testing existing .mo files in the wild.

The hash algorithm is implemented for MO files, this should result in faster access of the MO file. The hash is optional for Gettext and is not needed for reading or writing MO files, in this implementation it is always on and does produce sometimes different results to Gettext in very small files.

```
class translate.storage.mo.mofile(inputfile=None, **kwargs)
   A class representing a .mo file.

UnitClass
        alias of mounit

add_unit_to_index(unit)
        Add a unit to source and location idexes

addsourceunit(source)
        Add and returns a new unit with the given source string.
```

Return type TranslationUnit

#### addunit (unit)

Append the given unit to the object's list of units.

This method should always be used rather than trying to modify the list manually.

Parameters unit (TranslationUnit) - The unit that will be added.

### detect\_encoding (text, default\_encodings=None)

Try to detect a file encoding from *text*, using either the chardet lib or by trying to decode the file.

# fallback\_detection (text)

Simple detection based on BOM in case chardet is not available.

# findid(id)

find unit with matching id by checking id\_index

# findunit (source)

Find the unit with the given source string.

Return type TranslationUnit or None

### findunits (source)

Find the units with the given source string.

Return type TranslationUnit or None

# getheaderplural()

Returns the nplural and plural values from the header.

# getids (filename=None)

return a list of unit ids

# getprojectstyle()

Return the project based on information in the header.

# The project is determined in the following sequence:

- 1. Use the 'X-Project-Style' entry in the header.
- 2. Use 'Report-Msgid-Bug-To' entry
- 3. Use the 'X-Accelerator' entry
- 4. Use the Project ID
- 5. Analyse the file itself (not yet implemented)

# getsourcelanguage()

Get the source language for this store.

#### gettargetlanguage()

Return the target language based on information in the header.

# The target language is determined in the following sequence:

- 1. Use the 'Language' entry in the header.
- 2. Poedit's custom headers.
- 3. Analysing the 'Language-Team' entry.

### getunits()

Return a list of all units in this store.

# header()

Returns the header element, or None. Only the first element is allowed to be a header. Note that this could still return an empty header element, if present.

```
init_headers (charset='UTF-8', encoding='8bit', **kwargs)
     sets default values for po headers
isempty()
     Return True if the object doesn't contain any translation units.
makeheader (**kwargs)
     Create a header for the given filename.
     Check .makeheaderdict() for information on parameters.
makeheaderdict (charset='CHARSET',
                                             encoding='ENCODING',
                                                                          project_id_version=None,
                    pot_creation_date=None,
                                                po_revision_date=None,
                                                                             last_translator=None,
                                                                        plural_forms=None,
                    language team=None,
                                              mime_version=None,
                     port msgid bugs to=None, **kwargs)
     Create a header dictionary with useful defaults.
     pot_creation_date can be None (current date) or a value (datetime or string) po_revision_date can be None
     (form), False (=pot_creation_date), True (=now), or a value (datetime or string)
         Returns Dictionary with the header items
         Return type dict of strings
makeindex()
     Indexes the items in this store. At least .sourceindex should be useful.
merge on
     The matching criterion to use when merging on.
         Returns The default matching criterion for all the subclasses.
         Return type string
mergeheaders (otherstore)
     Merges another header with this header.
     This header is assumed to be the template.
parse (input)
     parses the given file or file source string
classmethod parsefile(storefile)
     Reads the given file (or opens the given filename) and parses back to an object.
parseheader()
     Parses the PO header and returns the interpreted values as a dictionary.
classmethod parsestring(storestring)
     Convert the string representation back to an object.
remove_unit_from_index(unit)
     Remove a unit from source and locaton indexes
require_index()
     make sure source index exists
```

save()

savefile (storefile)

Save to the file that data was originally read from, if available.

Write the string representation to the given file (or filename).

#### serialize(out)

Output a string representation of the MO data file

# setprojectstyle (project\_style)

Set the project in the header.

Parameters project\_style (str) - the new project

# setsourcelanguage (sourcelanguage)

Set the source language for this store.

# settargetlanguage (lang)

Set the target language in the header.

This removes any custom Poedit headers if they exist.

**Parameters** lang (str) – the new target language code

### translate(source)

Return the translated string for a given source string.

**Return type** String or None

### unit iter()

Iterator over all the units in this store.

# updatecontributor (name, email=None)

Add contribution comments if necessary.

# updateheader (add=False, \*\*kwargs)

Updates the fields in the PO style header.

This will create a header if add == True.

# updateheaderplural (nplurals, plural)

Update the Plural-Form PO header.

# class translate.storage.mo.mounit(source=None, \*\*kwargs)

A class representing a .mo translation message.

# adderror (errorname, errortext)

Adds an error message to this unit.

## **Parameters**

- **errorname** (*string*) A single word to id the error.
- **errortext** (*string*) The text describing the error.

### addlocation (location)

Add one location to the list of locations.

**Note:** Shouldn't be implemented if the format doesn't support it.

# addlocations (location)

Add a location or a list of locations.

**Note:** Most classes shouldn't need to implement this, but should rather implement TranslationUnit.addlocation().

**Warning:** This method might be removed in future.

# addnote (text, origin=None, position='append')

Adds a note (comment).

### **Parameters**

- **text** (*string*) Usually just a sentence or two.
- **origin** (*string*) Specifies who/where the comment comes from. Origin can be one of the following text strings: 'translator' 'developer', 'programmer', 'source code' (synonyms)

# classmethod buildfromunit (unit)

Build a native unit from a foreign unit, preserving as much information as possible.

### getcontext()

Get the message context

### geterrors()

Get all error messages.

# **Return type** Dictionary

## getid()

A unique identifier for this unit.

### Return type string

**Returns** an identifier for this unit that is unique in the store

Derived classes should override this in a way that guarantees a unique identifier for each unit in the store.

### getlocations()

A list of source code locations.

## **Return type** List

Note: Shouldn't be implemented if the format doesn't support it.

# getnotes (origin=None)

Returns all notes about this unit.

It will probably be freeform text or something reasonable that can be synthesised by the format. It should not include location comments (see <code>getlocations()</code>).

## gettargetlen()

Returns the length of the target string.

# Return type Integer

**Note:** Plural forms might be combined.

### getunits()

This unit in a list.

# hasplural()

Tells whether or not this specific unit has plural strings.

#### infer state()

Empty method that should be overridden in sub-classes to infer the current state(\_n) of the unit from its current state.

### isblank()

Used to see if this unit has no source or target string.

**Note:** This is probably used more to find translatable units, and we might want to move in that direction rather and get rid of this.

### isfuzzy()

Indicates whether this unit is fuzzy.

#### isheader()

Is this a header entry?

### isobsolete()

indicate whether a unit is obsolete

### isreview()

Indicates whether this unit needs review.

### istranslatable()

Is this message translateable?

#### istranslated()

Indicates whether this unit is translated.

This should be used rather than deducing it from .target, to ensure that other classes can implement more functionality (as XLIFF does).

# makeobsolete()

Make a unit obsolete

## markfuzzy (value=True)

Marks the unit as fuzzy or not.

# markreviewneeded (needsreview=True, explanation=None)

Marks the unit to indicate whether it needs review.

#### **Parameters**

- needsreview Defaults to True.
- **explanation** Adds an optional explanation as a note.

**merge** (otherunit, overwrite=False, comments=True, authoritative=False)

Do basic format agnostic merging.

# multistring\_to\_rich (mulstring)

Convert a multistring to a list of "rich" string trees:

# removenotes (origin=None)

Remove all the translator's notes.

```
rich source
          See also:
          rich_to_multistring(), multistring_to_rich()
     rich target
          See also:
          rich_to_multistring(), multistring_to_rich()
     classmethod rich_to_multistring(elem_list)
          Convert a "rich" string tree to a multistring:
          >>> from translate.storage.placeables.interfaces import X
          >>> rich = [StringElem(['foo', X(id='xxx', sub=[' ']), 'bar'])]
          >>> TranslationUnit.rich_to_multistring(rich)
          multistring(u'foo bar')
     setcontext(context)
          Set the message context
     setid(value)
          Sets the unique identified for this unit.
          only implemented if format allows ids independant from other unit properties like source or context
     unit iter()
          Iterator that only returns this unit.
translate.storage.mo.mounpack(filename='messages.mo')
     Helper to unpack Gettext MO files into a Python string
mozilla_lang
A class to manage Mozilla .lang files.
See https://github.com/mozilla-110n/langchecker/wiki/.lang-files-format for specifications on the format.
class translate.storage.mozilla_lang.LangStore(inputfile=None,
                                                                               mark active=False,
                                                             **kwargs)
     We extend TxtFile, since that has a lot of useful stuff for encoding
     UnitClass
          alias of LangUnit
     add_unit_to_index(unit)
          Add a unit to source and location idexes
     addsourceunit (source)
          Add and returns a new unit with the given source string.
              Return type TranslationUnit
     addunit (unit)
          Append the given unit to the object's list of units.
          This method should always be used rather than trying to modify the list manually.
              Parameters unit (TranslationUnit) - The unit that will be added.
```

```
detect_encoding(text, default_encodings=None)
     Try to detect a file encoding from text, using either the chardet lib or by trying to decode the file.
fallback_detection (text)
     Simple detection based on BOM in case chardet is not available.
findid(id)
     find unit with matching id by checking id index
findunit (source)
     Find the unit with the given source string.
         Return type TranslationUnit or None
findunits (source)
     Find the units with the given source string.
         Return type TranslationUnit or None
getids (filename=None)
     return a list of unit ids
getprojectstyle()
     Get the project type for this store.
getsourcelanguage()
     Get the source language for this store.
gettargetlanguage()
     Get the target language for this store.
getunits()
     Return a list of all units in this store.
isempty()
     Return True if the object doesn't contain any translation units.
makeindex()
     Indexes the items in this store. At least .sourceindex should be useful.
merge_on
     The matching criterion to use when merging on.
         Returns The default matching criterion for all the subclasses.
         Return type string
parse (lines)
     Read in text lines and create txtunits from the blocks of text
classmethod parsefile (storefile)
     Reads the given file (or opens the given filename) and parses back to an object.
classmethod parsestring(storestring)
     Convert the string representation back to an object.
remove_unit_from_index(unit)
     Remove a unit from source and locaton indexes
require_index()
     make sure source index exists
```

4.1. API 483

Save to the file that data was originally read from, if available.

save()

#### savefile (storefile)

Write the string representation to the given file (or filename).

### serialize(out)

Converts to a bytes representation that can be parsed back using parsestring(). out should be an open file-like objects to write to.

# setprojectstyle (project\_style)

Set the project type for this store.

# setsourcelanguage (sourcelanguage)

Set the source language for this store.

### settargetlanguage (targetlanguage)

Set the target language for this store.

# translate(source)

Return the translated string for a given source string.

Return type String or None

### unit iter()

Iterator over all the units in this store.

# class translate.storage.mozilla\_lang.LangUnit(source=None)

This is just a normal unit with a weird string output

#### adderror (errorname, errortext)

Adds an error message to this unit.

#### **Parameters**

- **errorname** (*string*) A single word to id the error.
- **errortext** (*string*) The text describing the error.

# addlocation (location)

Add one location to the list of locations.

**Note:** Shouldn't be implemented if the format doesn't support it.

# addlocations (location)

Add a location or a list of locations.

**Note:** Most classes shouldn't need to implement this, but should rather implement TranslationUnit.addlocation().

**Warning:** This method might be removed in future.

# addnote (text, origin=None, position='append')

Adds a note (comment).

#### **Parameters**

• **text** (*string*) – Usually just a sentence or two.

• **origin** (string) – Specifies who/where the comment comes from. Origin can be one of the following text strings: - 'translator' - 'developer', 'programmer', 'source code' (synonyms)

# classmethod buildfromunit (unit)

Build a native unit from a foreign unit, preserving as much information as possible.

### getcontext()

Get the message context.

# geterrors()

Get all error messages.

## **Return type** Dictionary

### getid()

A unique identifier for this unit.

# Return type string

**Returns** an identifier for this unit that is unique in the store

Derived classes should override this in a way that guarantees a unique identifier for each unit in the store.

### getlocations()

A list of source code locations.

# Return type List

**Note:** Shouldn't be implemented if the format doesn't support it.

# getnotes (origin=None)

Returns all notes about this unit.

It will probably be freeform text or something reasonable that can be synthesised by the format. It should not include location comments (see getlocations()).

### gettargetlen()

Returns the length of the target string.

# Return type Integer

**Note:** Plural forms might be combined.

#### getunits()

This unit in a list.

### hasplural()

Tells whether or not this specific unit has plural strings.

# infer\_state()

Empty method that should be overridden in sub-classes to infer the current state(\_n) of the unit from its current state.

### isblank()

Used to see if this unit has no source or target string.

**Note:** This is probably used more to find translatable units, and we might want to move in that direction rather and get rid of this.

# isfuzzy()

Indicates whether this unit is fuzzy.

### isheader()

Indicates whether this unit is a header.

#### isobsolete()

indicate whether a unit is obsolete

### isreview()

Indicates whether this unit needs review.

### istranslatable()

Indicates whether this unit can be translated.

This should be used to distinguish real units for translation from header, obsolete, binary or other blank units.

#### istranslated()

Indicates whether this unit is translated.

This should be used rather than deducing it from .target, to ensure that other classes can implement more functionality (as XLIFF does).

#### makeobsolete()

Make a unit obsolete

# markfuzzy(value=True)

Marks the unit as fuzzy or not.

# $\verb|markreviewneeded| (needs review=True, explanation=None)$

Marks the unit to indicate whether it needs review.

#### **Parameters**

- needsreview Defaults to True.
- **explanation** Adds an optional explanation as a note.

 $\textbf{merge} \ (other unit, overwrite = False, comments = True, authoritative = False)$ 

Do basic format agnostic merging.

## multistring\_to\_rich (mulstring)

Convert a multistring to a list of "rich" string trees:

## removenotes (origin=None)

Remove all the translator's notes.

# rich\_source

See also:

```
rich to multistring(), multistring to rich()
     rich_target
          See also:
          rich to multistring(), multistring to rich()
     classmethod rich_to_multistring(elem_list)
          Convert a "rich" string tree to a multistring:
          >>> from translate.storage.placeables.interfaces import X
          >>> rich = [StringElem(['foo', X(id='xxx', sub=[' ']), 'bar'])]
          >>> TranslationUnit.rich_to_multistring(rich)
          multistring(u'foo bar')
     setcontext (context)
          Set the message context
     setid(value)
          Sets the unique identified for this unit.
          only implemented if format allows ids independant from other unit properties like source or context
     unit iter()
          Iterator that only returns this unit.
odf_io
odf shared
omegat
Manage the OmegaT glossary format
OmegaT glossary format is used by the OmegaT computer aided translation tool.
It is a bilingual base class derived format with OmegaTFile and OmegaTUnit providing file and unit level access.
Format Implementation The OmegaT glossary format is a simple Tab Separated Value (TSV) file with the columns:
     source, target, comment.
     The dialect of the TSV files is specified by OmegaTDialect.
Encoding The files are either UTF-8 or encoded using the system default. UTF-8 encoded files use the .utf8 extension
     while system encoded files use the .tab extension.
translate.storage.omegat.OMEGAT_FIELDNAMES = [u'source', u'target', u'comment']
     Field names for an OmegaT glossary unit
class translate.storage.omegat.OmegaTDialect
     Describe the properties of an OmegaT generated TAB-delimited glossary file.
     classmethod combine (dialect, fmtparams)
```

4.1. API 487

Create a new dialect with defaults and added parameters.

An OmegaT glossary file

class translate.storage.omegat.OmegaTFile(inputfile=None, \*\*kwargs)

#### UnitClass

alias of OmegaTUnit

# add\_unit\_to\_index(unit)

Add a unit to source and location idexes

## addsourceunit (source)

Add and returns a new unit with the given source string.

Return type TranslationUnit

## addunit (unit)

Append the given unit to the object's list of units.

This method should always be used rather than trying to modify the list manually.

Parameters unit (TranslationUnit) - The unit that will be added.

# detect\_encoding(text, default\_encodings=None)

Try to detect a file encoding from text, using either the chardet lib or by trying to decode the file.

### fallback\_detection (text)

Simple detection based on BOM in case chardet is not available.

## findid(id)

find unit with matching id by checking id\_index

# findunit (source)

Find the unit with the given source string.

Return type TranslationUnit or None

### findunits (source)

Find the units with the given source string.

Return type TranslationUnit or None

# getids (filename=None)

return a list of unit ids

# getprojectstyle()

Get the project type for this store.

# getsourcelanguage()

Get the source language for this store.

# gettargetlanguage()

Get the target language for this store.

#### getunits()

Return a list of all units in this store.

# isempty()

Return True if the object doesn't contain any translation units.

## makeindex()

Indexes the items in this store. At least .sourceindex should be useful.

#### merge on

The matching criterion to use when merging on.

**Returns** The default matching criterion for all the subclasses.

Return type string

```
parse (input)
          parsese the given file or file source string
     classmethod parsefile(storefile)
          Reads the given file (or opens the given filename) and parses back to an object.
     classmethod parsestring(storestring)
          Convert the string representation back to an object.
     remove unit from index(unit)
          Remove a unit from source and locaton indexes
     require_index()
          make sure source index exists
     save()
          Save to the file that data was originally read from, if available.
     savefile (storefile)
          Write the string representation to the given file (or filename).
     serialize(out)
          Converts to a bytes representation that can be parsed back using parsestring(). out should be an
          open file-like objects to write to.
     setprojectstyle (project_style)
          Set the project type for this store.
     setsourcelanguage (sourcelanguage)
          Set the source language for this store.
     settargetlanguage (targetlanguage)
          Set the target language for this store.
     translate(source)
          Return the translated string for a given source string.
               Return type String or None
     unit_iter()
          Iterator over all the units in this store.
class translate.storage.omegat.OmegaTFileTab (inputfile=None, **kwargs)
     An OmegaT glossary file in the default system encoding
     UnitClass
          alias of OmegaTUnit
     add unit to index (unit)
          Add a unit to source and location idexes
     addsourceunit (source)
          Add and returns a new unit with the given source string.
               Return type TranslationUnit
     addunit (unit)
          Append the given unit to the object's list of units.
          This method should always be used rather than trying to modify the list manually.
               Parameters unit (TranslationUnit) – The unit that will be added.
     detect_encoding(text, default_encodings=None)
```

4.1. API 489

Try to detect a file encoding from text, using either the chardet lib or by trying to decode the file.

```
fallback detection (text)
     Simple detection based on BOM in case chardet is not available.
findid(id)
     find unit with matching id by checking id_index
findunit (source)
     Find the unit with the given source string.
         Return type TranslationUnit or None
findunits (source)
     Find the units with the given source string.
         Return type TranslationUnit or None
getids (filename=None)
     return a list of unit ids
getprojectstyle()
     Get the project type for this store.
getsourcelanguage()
     Get the source language for this store.
gettargetlanguage()
     Get the target language for this store.
getunits()
     Return a list of all units in this store.
isempty()
     Return True if the object doesn't contain any translation units.
makeindex()
     Indexes the items in this store. At least .sourceindex should be useful.
merge_on
     The matching criterion to use when merging on.
         Returns The default matching criterion for all the subclasses.
         Return type string
parse (input)
     parsese the given file or file source string
classmethod parsefile(storefile)
     Reads the given file (or opens the given filename) and parses back to an object.
classmethod parsestring(storestring)
     Convert the string representation back to an object.
remove_unit_from_index(unit)
     Remove a unit from source and locaton indexes
require_index()
     make sure source index exists
save()
     Save to the file that data was originally read from, if available.
savefile (storefile)
     Write the string representation to the given file (or filename).
```

#### serialize(out)

Converts to a bytes representation that can be parsed back using parsestring(). out should be an open file-like objects to write to.

# setprojectstyle (project\_style)

Set the project type for this store.

### setsourcelanguage (sourcelanguage)

Set the source language for this store.

# settargetlanguage (targetlanguage)

Set the target language for this store.

### translate(source)

Return the translated string for a given source string.

### Return type String or None

### unit\_iter()

Iterator over all the units in this store.

## class translate.storage.omegat.OmegaTUnit(source=None)

An OmegaT glossary unit

# adderror (errorname, errortext)

Adds an error message to this unit.

### **Parameters**

- **errorname** (*string*) A single word to id the error.
- **errortext** (*string*) The text describing the error.

# addlocation (location)

Add one location to the list of locations.

**Note:** Shouldn't be implemented if the format doesn't support it.

## addlocations (location)

Add a location or a list of locations.

Note: Most classes shouldn't need to implement this, but should rather implement  ${\tt TranslationUnit.addlocation()}$ .

**Warning:** This method might be removed in future.

addnote (text, origin=None, position=u'append')

Adds a note (comment).

### **Parameters**

- text (string) Usually just a sentence or two.
- **origin** (*string*) Specifies who/where the comment comes from. Origin can be one of the following text strings: 'translator' 'developer', 'programmer', 'source code' (synonyms)

### classmethod buildfromunit (unit)

Build a native unit from a foreign unit, preserving as much information as possible.

#### dict

Get the dictionary of values for a OmegaT line

### getcontext()

Get the message context.

#### getdict()

Get the dictionary of values for a OmegaT line

### geterrors()

Get all error messages.

# Return type Dictionary

### getid()

A unique identifier for this unit.

## Return type string

**Returns** an identifier for this unit that is unique in the store

Derived classes should override this in a way that guarantees a unique identifier for each unit in the store.

### getlocations()

A list of source code locations.

# Return type List

Note: Shouldn't be implemented if the format doesn't support it.

# getnotes (origin=None)

Returns all notes about this unit.

It will probably be freeform text or something reasonable that can be synthesised by the format. It should not include location comments (see getlocations()).

### gettargetlen()

Returns the length of the target string.

# Return type Integer

**Note:** Plural forms might be combined.

### getunits()

This unit in a list.

# hasplural()

Tells whether or not this specific unit has plural strings.

### infer\_state()

Empty method that should be overridden in sub-classes to infer the current state(\_n) of the unit from its current state.

# isblank()

Used to see if this unit has no source or target string.

**Note:** This is probably used more to find translatable units, and we might want to move in that direction rather and get rid of this.

# isfuzzy()

Indicates whether this unit is fuzzy.

### isheader()

Indicates whether this unit is a header.

#### isobsolete()

indicate whether a unit is obsolete

#### isreview()

Indicates whether this unit needs review.

### istranslatable()

Indicates whether this unit can be translated.

This should be used to distinguish real units for translation from header, obsolete, binary or other blank units.

#### istranslated()

Indicates whether this unit is translated.

This should be used rather than deducing it from .target, to ensure that other classes can implement more functionality (as XLIFF does).

#### makeobsolete()

Make a unit obsolete

# markfuzzy(value=True)

Marks the unit as fuzzy or not.

# $\verb|markreviewneeded| (needs review=True, explanation=None)$

Marks the unit to indicate whether it needs review.

### **Parameters**

- needsreview Defaults to True.
- **explanation** Adds an optional explanation as a note.

 $\textbf{merge} \ (other unit, overwrite = False, comments = True, authoritative = False)$ 

Do basic format agnostic merging.

## multistring\_to\_rich (mulstring)

Convert a multistring to a list of "rich" string trees:

## removenotes (origin=None)

Remove all the translator's notes.

# rich\_source

See also:

```
rich_to_multistring(), multistring_to_rich()
     rich_target
          See also:
          rich to multistring(), multistring to rich()
     classmethod rich_to_multistring(elem_list)
          Convert a "rich" string tree to a multistring:
          >>> from translate.storage.placeables.interfaces import X
          >>> rich = [StringElem(['foo', X(id='xxx', sub=[' ']), 'bar'])]
          >>> TranslationUnit.rich_to_multistring(rich)
          multistring(u'foo bar')
     setcontext (context)
          Set the message context
     setdict (newdict)
          Set the dictionary of values for a OmegaT line
              Parameters newdict (Dict) – a new dictionary with OmegaT line elements
     setid(value)
          Sets the unique identified for this unit.
          only implemented if format allows ids independant from other unit properties like source or context
     unit iter()
          Iterator that only returns this unit.
Classes that hold units of .oo files (oounit) or entire files (oofile).
These are specific .oo files for localisation exported by OpenOffice.org - SDF format (previously knows as GSI files).
The behaviour in terms of escaping is explained in detail in the programming comments.
translate.storage.oo.escape_help_text(text)
     Escapes the help text as it would be in an SDF file.
     <, >, " are only escaped in <[[:lower:]]> tags. Some HTML tags make it in in lowercase so those are dealt with.
     Some OpenOffice.org help tags are not escaped.
translate.storage.oo.escape_text(text)
     Escapes SDF text to be suitable for unit consumption.
translate.storage.oo.makekey(ookey, long_keys)
     converts an oo key tuple into a unique identifier
          Parameters
                • ookey (tuple) - an oo key
                • long_keys (Boolean) – Use long keys
```

Return type str

Returns unique ascii identifier

00

```
translate.storage.oo.normalizefilename(filename)
     converts any non-alphanumeric (standard roman) characters to _
class translate.storage.oo.oofile(input=None)
     this represents an entire .oo file
     UnitClass
          alias of oounit
     addline (thisline)
           adds a parsed line to the file
     getoutput (skip_source=False, fallback_lang=None)
           converts all the lines back to tab-delimited form
     parse (input)
           parses lines and adds them to the file
     serialize(out, skip_source=False, fallback_lang=None)
           convert to a string. double check that unicode is handled
class translate.storage.oo.ooline(parts=None)
     this represents one line, one translation in an .oo file
     getkey()
          get the key that identifies the resource
     getoutput()
          return a line in tab-delimited form
     getparts()
          return a list of parts in this line
     gettext()
           Obtains the text column and handle escaping.
     setparts (parts)
           create a line from its tab-delimited parts
     settext(text)
           Sets the text column and handle escaping.
     text
           Obtains the text column and handle escaping.
class translate.storage.oo.oomultifile (filename, mode=None, multifilestyle='single')
     this takes a huge GSI file and represents it as multiple smaller files. . .
     createsubfileindex()
           reads in all the lines and works out the subfiles
     getoofile (subfile)
           returns an oofile built up from the given subfile's lines
     getsubfilename (line)
          looks up the subfile name for the line
     getsubfilesrc(subfile)
           returns the list of lines matching the subfile
     listsubfiles()
           returns a list of subfiles in the file
```

```
openinputfile (subfile)
            returns a pseudo-file object for the given subfile
      openoutputfile (subfile)
            returns a pseudo-file object for the given subfile
class translate.storage.oo.oounit
      this represents a number of translations of a resource
      addline (line)
            add a line to the oounit
      getoutput (skip_source=False, fallback_lang=None)
            return the lines in tab-delimited form
translate.storage.oo.unescape_help_text(text)
      Unescapes normal text to be suitable for writing to the SDF file.
translate.storage.oo.unescape_text(text)
      Unescapes SDF text to be suitable for unit consumption.
class translate.storage.oo.unormalizechar(normalchars)
      clear() \rightarrow None. Remove all items from D.
      copy() \rightarrow a \text{ shallow copy of } D
      fromkeys (S|, v|) \rightarrow \text{New dict with keys from S and values equal to v.}
            v defaults to None.
      get(k|,d|) \rightarrow D[k] if k in D, else d. d defaults to None.
      has_key (k) \rightarrow True if D has a key k, else False
      items () \rightarrow list of D's (key, value) pairs, as 2-tuples
      iteritems () \rightarrow an iterator over the (key, value) items of D
      iterkeys () \rightarrow an iterator over the keys of D
      \textbf{itervalues} \ () \ \rightarrow \text{an iterator over the values of } D
      keys () \rightarrow list of D's keys
      pop(k|,d|) \rightarrow v, remove specified key and return the corresponding value.
            If key is not found, d is returned if given, otherwise KeyError is raised
      popitem () \rightarrow (k, v), remove and return some (key, value) pair as a
            2-tuple; but raise KeyError if D is empty.
      setdefault (k[,d]) \rightarrow D.get(k,d), also set D[k]=d if k not in D
      update ([E], **F) \rightarrow None. Update D from dict/iterable E and F.
            If E present and has a .keys() method, does: for k in E: D[k] = E[k] If E present and lacks .keys() method,
            does: for (k, v) in E: D[k] = v In either case, this is followed by: for k in F: D[k] = F[k]
      values () \rightarrow list of D's values
      viewitems () \rightarrow a set-like object providing a view on D's items
      viewkeys () \rightarrow a set-like object providing a view on D's keys
      viewvalues () \rightarrow an object providing a view on D's values
```

# placeables

This module implements basic functionality to support placeables.

# A placeable is used to represent things like:

### 1. Substitutions

For example, in ODF, footnotes appear in the ODF XML where they are defined; so if we extract a paragraph with some footnotes, the translator will have a lot of additional XML to with; so we separate the footnotes out into separate translation units and mark their positions in the original text with placeables.

2. Hiding of inline formatting data

The translator doesn't want to have to deal with all the weird formatting conventions of wherever the text came from.

3. Marking variables

This is an old issue - translators translate variable names which should remain untranslated. We can wrap placeables around variable names to avoid this.

The placeables model follows the XLIFF standard's list of placeables. Please refer to the XLIFF specification to get a better understanding.

#### base

Contains base placeable classes with names based on XLIFF placeables. See the XLIFF standard for more information about what the names mean.

```
apply_to_strings(f)
```

Apply f to all actual strings in the tree.

**Parameters f** – Must take one (str or unicode) argument and return a string or unicode.

copy()

Returns a copy of the sub-tree. This should be overridden in sub-classes with more data.

Note: self.renderer is not copied.

```
delete_range (start_index, end_index)
```

Delete the text in the range given by the string-indexes start\_index and end\_index.

Partial nodes will only be removed if they are editable.

Returns A StringElem representing the removed sub-string, the parent node from which it was deleted as well as the offset at which it was deleted from. None is returned for the parent value if the root was deleted. If the parent and offset values are not None, parent. insert (offset, deleted) effectively undoes the delete.

```
depth_first (filter=None)
```

Returns a list of the nodes in the tree in depth-first order.

### elem\_at\_offset (offset)

Get the StringElem in the tree that contains the string rendered at the given offset.

#### elem offset (elem)

Find the offset of elem in the current tree.

This cannot be reliably used if self.renderer is used and even less so if the rendering function renders the string differently upon different calls. In Virtual the StringElemGUI.index() method is used as replacement for this one.

**Returns** The string index where element e starts, or -1 if e was not found.

#### encode (encoding='ascii')

More unicode class emulation.

### find(x)

Find sub-string x in this string tree and return the position at which it starts.

# $find_elems_with(x)$

Find all elements in the current sub-tree containing x.

# flatten (filter=None)

Flatten the tree by returning a depth-first search over the tree's leaves.

### get index data(index)

Get info about the specified range in the tree.

#### Returns

A dictionary with the following items:

- elem: The element in which index resides.
- *index*: Copy of the index parameter
- offset: The offset of index into 'elem'.

# get\_parent\_elem(child)

Searches the current sub-tree for and returns the parent of the child element.

# insert (offset, text, preferred\_parent=None)

Insert the given text at the specified offset of this string-tree's string (Unicode) representation.

# insert\_between (left, right, text)

Insert the given text between the two parameter StringElems.

# isleaf()

Whether or not this instance is a leaf node in the StringElem tree.

A node is a leaf node if it is a StringElem (not a sub-class) and contains only sub-elements of type str or unicode.

### Return type bool

### iter\_depth\_first(filter=None)

Iterate through the nodes in the tree in dept-first order.

# map (f, filter=None)

Apply f to all nodes for which filter returned True (optional).

### print\_tree (indent=0, verbose=False)

Print the tree from the current instance's point in an indented manner.

### prune()

Remove unnecessary nodes to make the tree optimal.

# remove\_type (ptype)

Replace nodes with type ptype with base StringElems, containing the same sub-elements. This is only applicable to elements below the element tree root node.

### translate()

Transform the sub-tree according to some class-specific needs. This method should be either overridden in implementing sub-classes or dynamically replaced by specific applications.

**Returns** The transformed Unicode string representing the sub-tree.

## apply\_to\_strings(f)

Apply f to all actual strings in the tree.

**Parameters f** – Must take one (str or unicode) argument and return a string or unicode.

#### copy()

Returns a copy of the sub-tree. This should be overridden in sub-classes with more data.

**Note:** self.renderer is **not** copied.

### delete\_range (start\_index, end\_index)

Delete the text in the range given by the string-indexes start\_index and end\_index.

Partial nodes will only be removed if they are editable.

Returns A StringElem representing the removed sub-string, the parent node from which it was deleted as well as the offset at which it was deleted from. None is returned for the parent value if the root was deleted. If the parent and offset values are not None, parent. insert (offset, deleted) effectively undoes the delete.

# depth\_first (filter=None)

Returns a list of the nodes in the tree in depth-first order.

#### elem at offset(offset)

Get the StringElem in the tree that contains the string rendered at the given offset.

# $elem\_offset (elem)$

Find the offset of elem in the current tree.

This cannot be reliably used if self.renderer is used and even less so if the rendering function renders the string differently upon different calls. In Virtual the StringElemGUI.index() method is used as replacement for this one.

**Returns** The string index where element e starts, or -1 if e was not found.

# encode (encoding='ascii')

More unicode class emulation.

#### find(x)

Find sub-string x in this string tree and return the position at which it starts.

### $find_elems_with(x)$

Find all elements in the current sub-tree containing x.

# flatten (filter=None)

Flatten the tree by returning a depth-first search over the tree's leaves.

# get\_index\_data(index)

Get info about the specified range in the tree.

#### Returns

A dictionary with the following items:

- elem: The element in which index resides.
- index: Copy of the index parameter
- offset: The offset of index into 'elem'.

#### get parent elem (child)

Searches the current sub-tree for and returns the parent of the child element.

# insert (offset, text, preferred\_parent=None)

Insert the given text at the specified offset of this string-tree's string (Unicode) representation.

# insert\_between (left, right, text)

Insert the given text between the two parameter StringElems.

### isleaf()

Whether or not this instance is a leaf node in the StringElem tree.

A node is a leaf node if it is a StringElem (not a sub-class) and contains only sub-elements of type str or unicode.

# Return type bool

# iter\_depth\_first (filter=None)

Iterate through the nodes in the tree in dept-first order.

# map (f, filter=None)

Apply f to all nodes for which filter returned True (optional).

## print\_tree (indent=0, verbose=False)

Print the tree from the current instance's point in an indented manner.

# prune()

Remove unnecessary nodes to make the tree optimal.

# remove\_type (ptype)

Replace nodes with type ptype with base StringElems, containing the same sub-elements. This is only applicable to elements below the element tree root node.

# translate()

Transform the sub-tree according to some class-specific needs. This method should be either overridden in implementing sub-classes or dynamically replaced by specific applications.

**Returns** The transformed Unicode string representing the sub-tree.

class translate.storage.placeables.base.Ph(sub=None, id=None, rid=None, xid=None, xid=None, \*\*kwargs)

### apply\_to\_strings(f)

Apply f to all actual strings in the tree.

**Parameters f** – Must take one (str or unicode) argument and return a string or unicode.

# copy()

Returns a copy of the sub-tree. This should be overridden in sub-classes with more data.

**Note:** self.renderer is **not** copied.

### **delete range** (*start index*, *end index*)

Delete the text in the range given by the string-indexes start\_index and end\_index.

Partial nodes will only be removed if they are editable.

Returns A StringElem representing the removed sub-string, the parent node from which it was deleted as well as the offset at which it was deleted from. None is returned for the parent value if the root was deleted. If the parent and offset values are not None, parent. insert (offset, deleted) effectively undoes the delete.

## depth\_first (filter=None)

Returns a list of the nodes in the tree in depth-first order.

### elem\_at\_offset (offset)

Get the StringElem in the tree that contains the string rendered at the given offset.

#### elem offset (elem)

Find the offset of elem in the current tree.

This cannot be reliably used if self.renderer is used and even less so if the rendering function renders the string differently upon different calls. In Virtual the StringElemGUI.index() method is used as replacement for this one.

**Returns** The string index where element e starts, or -1 if e was not found.

### encode (encoding='ascii')

More unicode class emulation.

#### find(x)

Find sub-string x in this string tree and return the position at which it starts.

### $find_elems_with(x)$

Find all elements in the current sub-tree containing x.

## flatten (filter=None)

Flatten the tree by returning a depth-first search over the tree's leaves.

## get\_index\_data(index)

Get info about the specified range in the tree.

# Returns

A dictionary with the following items:

- *elem*: The element in which index resides.
- index: Copy of the index parameter
- offset: The offset of index into 'elem'.

#### get\_parent\_elem(child)

Searches the current sub-tree for and returns the parent of the child element.

## insert (offset, text, preferred\_parent=None)

Insert the given text at the specified offset of this string-tree's string (Unicode) representation.

### insert\_between (left, right, text)

Insert the given text between the two parameter StringElems.

## isleaf()

Whether or not this instance is a leaf node in the StringElem tree.

A node is a leaf node if it is a StringElem (not a sub-class) and contains only sub-elements of type str or unicode.

### Return type bool

## iter\_depth\_first (filter=None)

Iterate through the nodes in the tree in dept-first order.

## map (f, filter=None)

Apply f to all nodes for which filter returned True (optional).

### print tree (indent=0, verbose=False)

Print the tree from the current instance's point in an indented manner.

## prune()

Remove unnecessary nodes to make the tree optimal.

# remove\_type (ptype)

Replace nodes with type ptype with base StringElems, containing the same sub-elements. This is only applicable to elements below the element tree root node.

## translate()

Transform the sub-tree according to some class-specific needs. This method should be either overridden in implementing sub-classes or dynamically replaced by specific applications.

**Returns** The transformed Unicode string representing the sub-tree.

## $apply_to_strings(f)$

Apply f to all actual strings in the tree.

**Parameters f** – Must take one (str or unicode) argument and return a string or unicode.

### copy()

Returns a copy of the sub-tree. This should be overridden in sub-classes with more data.

Note: self.renderer is not copied.

## delete\_range (start\_index, end\_index)

Delete the text in the range given by the string-indexes start\_index and end\_index.

Partial nodes will only be removed if they are editable.

**Returns** A StringElem representing the removed sub-string, the parent node from which it was deleted as well as the offset at which it was deleted from. None is returned for the parent value if the root was deleted. If the parent and offset values are not None, parent. insert (offset, deleted) effectively undoes the delete.

### depth\_first (filter=None)

Returns a list of the nodes in the tree in depth-first order.

# elem\_at\_offset (offset)

Get the StringElem in the tree that contains the string rendered at the given offset.

### elem offset(elem)

Find the offset of elem in the current tree.

This cannot be reliably used if self.renderer is used and even less so if the rendering function renders the string differently upon different calls. In Virtual the StringElemGUI.index() method is used as replacement for this one.

**Returns** The string index where element e starts, or -1 if e was not found.

### encode (encoding='ascii')

More unicode class emulation.

#### find(x)

Find sub-string x in this string tree and return the position at which it starts.

## $find_elems_with(x)$

Find all elements in the current sub-tree containing x.

### flatten (filter=None)

Flatten the tree by returning a depth-first search over the tree's leaves.

## get\_index\_data(index)

Get info about the specified range in the tree.

#### Returns

A dictionary with the following items:

- elem: The element in which index resides.
- index: Copy of the index parameter
- offset: The offset of index into 'elem'.

## get\_parent\_elem(child)

Searches the current sub-tree for and returns the parent of the child element.

## insert (offset, text, preferred\_parent=None)

Insert the given text at the specified offset of this string-tree's string (Unicode) representation.

## insert\_between (left, right, text)

Insert the given text between the two parameter StringElems.

# isleaf()

Whether or not this instance is a leaf node in the StringElem tree.

A node is a leaf node if it is a StringElem (not a sub-class) and contains only sub-elements of type str or unicode.

## Return type bool

### iter\_depth\_first (filter=None)

Iterate through the nodes in the tree in dept-first order.

## map(f, filter=None)

Apply f to all nodes for which filter returned True (optional).

## print\_tree (indent=0, verbose=False)

Print the tree from the current instance's point in an indented manner.

### prune()

Remove unnecessary nodes to make the tree optimal.

# $remove\_type(ptype)$

Replace nodes with type ptype with base StringElems, containing the same sub-elements. This is only applicable to elements below the element tree root node.

### translate()

Transform the sub-tree according to some class-specific needs. This method should be either overridden in implementing sub-classes or dynamically replaced by specific applications.

**Returns** The transformed Unicode string representing the sub-tree.

## $apply_to_strings(f)$

Apply f to all actual strings in the tree.

**Parameters f** – Must take one (str or unicode) argument and return a string or unicode.

#### copy()

Returns a copy of the sub-tree. This should be overridden in sub-classes with more data.

Note: self.renderer is not copied.

## delete\_range (start\_index, end\_index)

Delete the text in the range given by the string-indexes start\_index and end\_index.

Partial nodes will only be removed if they are editable.

**Returns** A StringElem representing the removed sub-string, the parent node from which it was deleted as well as the offset at which it was deleted from. None is returned for the parent value if the root was deleted. If the parent and offset values are not None, parent. insert (offset, deleted) effectively undoes the delete.

#### depth first(filter=None)

Returns a list of the nodes in the tree in depth-first order.

## elem\_at\_offset (offset)

Get the StringElem in the tree that contains the string rendered at the given offset.

#### elem\_offset (elem)

Find the offset of elem in the current tree.

This cannot be reliably used if self.renderer is used and even less so if the rendering function renders the string differently upon different calls. In Virtual the StringElemGUI.index() method is used as replacement for this one.

**Returns** The string index where element e starts, or -1 if e was not found.

### encode (encoding='ascii')

More unicode class emulation.

# find(x)

Find sub-string x in this string tree and return the position at which it starts.

## $find_elems_with(x)$

Find all elements in the current sub-tree containing x.

## flatten (filter=None)

Flatten the tree by returning a depth-first search over the tree's leaves.

# get\_index\_data(index)

Get info about the specified range in the tree.

### Returns

A dictionary with the following items:

- *elem*: The element in which index resides.
- index: Copy of the index parameter
- offset: The offset of index into 'elem'.

# get\_parent\_elem (child)

Searches the current sub-tree for and returns the parent of the child element.

## insert (offset, text, preferred\_parent=None)

Insert the given text at the specified offset of this string-tree's string (Unicode) representation.

### insert\_between (left, right, text)

Insert the given text between the two parameter StringElems.

#### isleaf()

Whether or not this instance is a leaf node in the StringElem tree.

A node is a leaf node if it is a StringElem (not a sub-class) and contains only sub-elements of type str or unicode.

## Return type bool

# iter\_depth\_first (filter=None)

Iterate through the nodes in the tree in dept-first order.

### map(f, filter=None)

Apply f to all nodes for which filter returned True (optional).

### print\_tree (indent=0, verbose=False)

Print the tree from the current instance's point in an indented manner.

### prune()

Remove unnecessary nodes to make the tree optimal.

#### remove\_type (ptype)

Replace nodes with type ptype with base StringElems, containing the same sub-elements. This is only applicable to elements below the element tree root node.

## translate()

Transform the sub-tree according to some class-specific needs. This method should be either overridden in implementing sub-classes or dynamically replaced by specific applications.

**Returns** The transformed Unicode string representing the sub-tree.

**class** translate.storage.placeables.base.**Bx**(id=None, xid=None, \*\*kwargs)

## apply\_to\_strings(f)

Apply f to all actual strings in the tree.

**Parameters f** – Must take one (str or unicode) argument and return a string or unicode.

## copy()

Returns a copy of the sub-tree. This should be overridden in sub-classes with more data.

Note: self.renderer is not copied.

## delete\_range (start\_index, end\_index)

Delete the text in the range given by the string-indexes start\_index and end\_index.

Partial nodes will only be removed if they are editable.

Returns A StringElem representing the removed sub-string, the parent node from which it was deleted as well as the offset at which it was deleted from. None is returned for the parent value if the root was deleted. If the parent and offset values are not None, parent. insert (offset, deleted) effectively undoes the delete.

### depth first(filter=None)

Returns a list of the nodes in the tree in depth-first order.

## elem\_at\_offset (offset)

Get the StringElem in the tree that contains the string rendered at the given offset.

# elem\_offset (elem)

Find the offset of elem in the current tree.

This cannot be reliably used if self.renderer is used and even less so if the rendering function renders the string differently upon different calls. In Virtual the StringElemGUI.index() method is used as replacement for this one.

**Returns** The string index where element e starts, or -1 if e was not found.

### encode (encoding='ascii')

More unicode class emulation.

### find(x)

Find sub-string x in this string tree and return the position at which it starts.

#### find elems with (x)

Find all elements in the current sub-tree containing x.

### flatten (filter=None)

Flatten the tree by returning a depth-first search over the tree's leaves.

### get\_index\_data(index)

Get info about the specified range in the tree.

### Returns

A dictionary with the following items:

- elem: The element in which index resides.
- index: Copy of the index parameter
- offset: The offset of index into 'elem'.

## get\_parent\_elem(child)

Searches the current sub-tree for and returns the parent of the child element.

# insert (offset, text, preferred\_parent=None)

Insert the given text at the specified offset of this string-tree's string (Unicode) representation.

## insert\_between (left, right, text)

Insert the given text between the two parameter StringElems.

#### isleaf()

Whether or not this instance is a leaf node in the StringElem tree.

A node is a leaf node if it is a StringElem (not a sub-class) and contains only sub-elements of type str or unicode.

## Return type bool

## iter\_depth\_first (filter=None)

Iterate through the nodes in the tree in dept-first order.

## map(f, filter=None)

Apply f to all nodes for which filter returned True (optional).

## print tree(indent=0, verbose=False)

Print the tree from the current instance's point in an indented manner.

### prune()

Remove unnecessary nodes to make the tree optimal.

# remove\_type (ptype)

Replace nodes with type ptype with base StringElems, containing the same sub-elements. This is only applicable to elements below the element tree root node.

#### translate()

Transform the sub-tree according to some class-specific needs. This method should be either overridden in implementing sub-classes or dynamically replaced by specific applications.

**Returns** The transformed Unicode string representing the sub-tree.

**class** translate.storage.placeables.base.**Ex** (*id=None*, *xid=None*, \*\*kwargs)

# ${\tt apply\_to\_strings}\,(f)$

Apply f to all actual strings in the tree.

**Parameters f** – Must take one (str or unicode) argument and return a string or unicode.

## copy()

Returns a copy of the sub-tree. This should be overridden in sub-classes with more data.

Note: self.renderer is not copied.

## delete\_range (start\_index, end\_index)

Delete the text in the range given by the string-indexes start index and end index.

Partial nodes will only be removed if they are editable.

Returns A StringElem representing the removed sub-string, the parent node from which it was deleted as well as the offset at which it was deleted from. None is returned for the parent value if the root was deleted. If the parent and offset values are not None, parent. insert (offset, deleted) effectively undoes the delete.

### depth\_first (filter=None)

Returns a list of the nodes in the tree in depth-first order.

## elem\_at\_offset (offset)

Get the StringElem in the tree that contains the string rendered at the given offset.

# ${\tt elem\_offset}~(elem)$

Find the offset of elem in the current tree.

This cannot be reliably used if self.renderer is used and even less so if the rendering function renders the string differently upon different calls. In Virtual the StringElemGUI.index() method is used as replacement for this one.

**Returns** The string index where element e starts, or -1 if e was not found.

# encode (encoding='ascii')

More unicode class emulation.

#### find(x)

Find sub-string x in this string tree and return the position at which it starts.

## $find_elems_with(x)$

Find all elements in the current sub-tree containing x.

### flatten (filter=None)

Flatten the tree by returning a depth-first search over the tree's leaves.

### get index data(index)

Get info about the specified range in the tree.

#### Returns

A dictionary with the following items:

- *elem*: The element in which index resides.
- *index*: Copy of the index parameter
- offset: The offset of index into 'elem'.

## get\_parent\_elem(child)

Searches the current sub-tree for and returns the parent of the child element.

## insert (offset, text, preferred\_parent=None)

Insert the given text at the specified offset of this string-tree's string (Unicode) representation.

## insert\_between (left, right, text)

Insert the given text between the two parameter StringElems.

#### isleaf(

Whether or not this instance is a leaf node in the StringElem tree.

A node is a leaf node if it is a StringElem (not a sub-class) and contains only sub-elements of type str or unicode.

### Return type bool

## iter\_depth\_first (filter=None)

Iterate through the nodes in the tree in dept-first order.

## map (f, filter=None)

Apply f to all nodes for which filter returned True (optional).

# print\_tree (indent=0, verbose=False)

Print the tree from the current instance's point in an indented manner.

### prune()

Remove unnecessary nodes to make the tree optimal.

### remove\_type (ptype)

Replace nodes with type ptype with base StringElems, containing the same sub-elements. This is only applicable to elements below the element tree root node.

### translate()

Transform the sub-tree according to some class-specific needs. This method should be either overridden in implementing sub-classes or dynamically replaced by specific applications.

**Returns** The transformed Unicode string representing the sub-tree.

class translate.storage.placeables.base.X(id=None, xid=None, \*\*kwargs)

## $apply_to_strings(f)$

Apply f to all actual strings in the tree.

**Parameters f** – Must take one (str or unicode) argument and return a string or unicode.

## copy()

Returns a copy of the sub-tree. This should be overridden in sub-classes with more data.

Note: self.renderer is not copied.

### delete\_range (start\_index, end\_index)

Delete the text in the range given by the string-indexes start\_index and end\_index.

Partial nodes will only be removed if they are editable.

**Returns** A StringElem representing the removed sub-string, the parent node from which it was deleted as well as the offset at which it was deleted from. None is returned for the parent value if the root was deleted. If the parent and offset values are not None, parent. insert (offset, deleted) effectively undoes the delete.

## depth\_first (filter=None)

Returns a list of the nodes in the tree in depth-first order.

### elem\_at\_offset (offset)

Get the StringElem in the tree that contains the string rendered at the given offset.

#### elem offset(elem)

Find the offset of elem in the current tree.

This cannot be reliably used if self.renderer is used and even less so if the rendering function renders the string differently upon different calls. In Virtual the StringElemGUI.index() method is used as replacement for this one.

**Returns** The string index where element e starts, or -1 if e was not found.

#### encode (encoding='ascii')

More unicode class emulation.

# find(x)

Find sub-string x in this string tree and return the position at which it starts.

## $find_elems_with(x)$

Find all elements in the current sub-tree containing x.

### flatten (filter=None)

Flatten the tree by returning a depth-first search over the tree's leaves.

## get\_index\_data(index)

Get info about the specified range in the tree.

#### Returns

A dictionary with the following items:

- elem: The element in which index resides.
- *index*: Copy of the index parameter
- offset: The offset of index into 'elem'.

# get\_parent\_elem (child)

Searches the current sub-tree for and returns the parent of the child element.

### insert (offset, text, preferred\_parent=None)

Insert the given text at the specified offset of this string-tree's string (Unicode) representation.

## insert\_between (left, right, text)

Insert the given text between the two parameter StringElems.

#### isleaf()

Whether or not this instance is a leaf node in the StringElem tree.

A node is a leaf node if it is a StringElem (not a sub-class) and contains only sub-elements of type str or unicode.

## Return type bool

## iter\_depth\_first (filter=None)

Iterate through the nodes in the tree in dept-first order.

### map (f, filter=None)

Apply f to all nodes for which filter returned True (optional).

### print\_tree (indent=0, verbose=False)

Print the tree from the current instance's point in an indented manner.

### prune()

Remove unnecessary nodes to make the tree optimal.

### remove\_type (ptype)

Replace nodes with type ptype with base StringElems, containing the same sub-elements. This is only applicable to elements below the element tree root node.

### translate()

Transform the sub-tree according to some class-specific needs. This method should be either overridden in implementing sub-classes or dynamically replaced by specific applications.

**Returns** The transformed Unicode string representing the sub-tree.

class translate.storage.placeables.base.Sub(sub=None, id=None, rid=None, xid=None, xid=None, rid=None, xid=None, xid=None

# apply\_to\_strings(f)

Apply f to all actual strings in the tree.

**Parameters f** – Must take one (str or unicode) argument and return a string or unicode.

#### copy()

Returns a copy of the sub-tree. This should be overridden in sub-classes with more data.

Note: self.renderer is not copied.

## delete range (start index, end index)

Delete the text in the range given by the string-indexes start\_index and end\_index.

Partial nodes will only be removed if they are editable.

Returns A StringElem representing the removed sub-string, the parent node from which it was deleted as well as the offset at which it was deleted from. None is returned for the parent value if the root was deleted. If the parent and offset values are not None, parent. insert (offset, deleted) effectively undoes the delete.

# depth\_first (filter=None)

Returns a list of the nodes in the tree in depth-first order.

### elem\_at\_offset (offset)

Get the StringElem in the tree that contains the string rendered at the given offset.

# ${\tt elem\_offset}\;(elem)$

Find the offset of elem in the current tree.

This cannot be reliably used if self.renderer is used and even less so if the rendering function renders the string differently upon different calls. In Virtual the StringElemGUI.index() method is used as replacement for this one.

**Returns** The string index where element e starts, or -1 if e was not found.

## encode (encoding='ascii')

More unicode class emulation.

#### find(x)

Find sub-string x in this string tree and return the position at which it starts.

### $find_elems_with(x)$

Find all elements in the current sub-tree containing x.

## flatten (filter=None)

Flatten the tree by returning a depth-first search over the tree's leaves.

## get\_index\_data(index)

Get info about the specified range in the tree.

#### **Returns**

A dictionary with the following items:

- elem: The element in which index resides.
- index: Copy of the index parameter
- offset: The offset of index into 'elem'.

# get\_parent\_elem(child)

Searches the current sub-tree for and returns the parent of the child element.

# insert (offset, text, preferred\_parent=None)

Insert the given text at the specified offset of this string-tree's string (Unicode) representation.

## insert\_between (left, right, text)

Insert the given text between the two parameter StringElems.

## isleaf()

Whether or not this instance is a leaf node in the StringElem tree.

A node is a leaf node if it is a StringElem (not a sub-class) and contains only sub-elements of type str or unicode.

# Return type bool

## iter\_depth\_first (filter=None)

Iterate through the nodes in the tree in dept-first order.

### map(f, filter=None)

Apply f to all nodes for which filter returned True (optional).

## print\_tree (indent=0, verbose=False)

Print the tree from the current instance's point in an indented manner.

### prune()

Remove unnecessary nodes to make the tree optimal.

## remove\_type (ptype)

Replace nodes with type ptype with base StringElems, containing the same sub-elements. This is only applicable to elements below the element tree root node.

#### translate()

Transform the sub-tree according to some class-specific needs. This method should be either overridden in implementing sub-classes or dynamically replaced by specific applications.

**Returns** The transformed Unicode string representing the sub-tree.

### general

Contains general placeable implementations. That is placeables that does not fit into any other sub-category.

Placeable for the "alt=..." attributes inside XML tags.

```
apply_to_strings(f)
```

Apply f to all actual strings in the tree.

**Parameters f** – Must take one (str or unicode) argument and return a string or unicode.

copy()

Returns a copy of the sub-tree. This should be overridden in sub-classes with more data.

Note: self.renderer is not copied.

```
delete_range (start_index, end_index)
```

Delete the text in the range given by the string-indexes start\_index and end\_index.

Partial nodes will only be removed if they are editable.

Returns A StringElem representing the removed sub-string, the parent node from which it was deleted as well as the offset at which it was deleted from. None is returned for the parent value if the root was deleted. If the parent and offset values are not None, parent. insert (offset, deleted) effectively undoes the delete.

```
depth_first (filter=None)
```

Returns a list of the nodes in the tree in depth-first order.

```
elem_at_offset (offset)
```

Get the StringElem in the tree that contains the string rendered at the given offset.

```
{\tt elem\_offset}\;(elem)
```

Find the offset of elem in the current tree.

This cannot be reliably used if self.renderer is used and even less so if the rendering function renders the string differently upon different calls. In Virtual the StringElemGUI.index() method is used as replacement for this one.

**Returns** The string index where element e starts, or -1 if e was not found.

```
encode (encoding='ascii')
```

More unicode class emulation.

#### find(x)

Find sub-string  $\mathbf{x}$  in this string tree and return the position at which it starts.

## $find_elems_with(x)$

Find all elements in the current sub-tree containing x.

### flatten (filter=None)

Flatten the tree by returning a depth-first search over the tree's leaves.

## get\_index\_data(index)

Get info about the specified range in the tree.

#### Returns

A dictionary with the following items:

- elem: The element in which index resides.
- *index*: Copy of the index parameter
- offset: The offset of index into 'elem'.

### get\_parent\_elem(child)

Searches the current sub-tree for and returns the parent of the child element.

## insert (offset, text, preferred\_parent=None)

Insert the given text at the specified offset of this string-tree's string (Unicode) representation.

### insert between (left, right, text)

Insert the given text between the two parameter StringElems.

## isleaf()

Whether or not this instance is a leaf node in the StringElem tree.

A node is a leaf node if it is a StringElem (not a sub-class) and contains only sub-elements of type str or unicode.

## Return type bool

## iter\_depth\_first (filter=None)

Iterate through the nodes in the tree in dept-first order.

```
map (f, filter=None)
```

Apply f to all nodes for which filter returned True (optional).

### classmethod parse(pstr)

A parser method to extract placeables from a string based on a regular expression. Use this function as the @parse() method of a placeable class.

# print\_tree (indent=0, verbose=False)

Print the tree from the current instance's point in an indented manner.

## prune()

Remove unnecessary nodes to make the tree optimal.

### remove\_type (ptype)

Replace nodes with type ptype with base StringElems, containing the same sub-elements. This is only applicable to elements below the element tree root node.

# translate()

Transform the sub-tree according to some class-specific needs. This method should be either overridden in implementing sub-classes or dynamically replaced by specific applications.

**Returns** The transformed Unicode string representing the sub-tree.

```
class translate.storage.placeables.general.XMLEntityPlaceable (sub=None, id=None, rid=None, xid=None, xid=None, **kwargs)
```

Placeable handling XML entities (&xxxxx; -style entities).

## apply\_to\_strings(f)

Apply f to all actual strings in the tree.

**Parameters f** – Must take one (str or unicode) argument and return a string or unicode.

### copy()

Returns a copy of the sub-tree. This should be overridden in sub-classes with more data.

Note: self.renderer is not copied.

## delete\_range (start\_index, end\_index)

Delete the text in the range given by the string-indexes start\_index and end\_index.

Partial nodes will only be removed if they are editable.

**Returns** A StringElem representing the removed sub-string, the parent node from which it was deleted as well as the offset at which it was deleted from. None is returned for the parent value if the root was deleted. If the parent and offset values are not None, parent. insert (offset, deleted) effectively undoes the delete.

### depth\_first (filter=None)

Returns a list of the nodes in the tree in depth-first order.

### elem at offset(offset)

Get the StringElem in the tree that contains the string rendered at the given offset.

#### elem offset(elem)

Find the offset of elem in the current tree.

This cannot be reliably used if self.renderer is used and even less so if the rendering function renders the string differently upon different calls. In Virtual the StringElemGUI.index() method is used as replacement for this one.

**Returns** The string index where element e starts, or -1 if e was not found.

## encode (encoding='ascii')

More unicode class emulation.

## find(x)

Find sub-string x in this string tree and return the position at which it starts.

## $find_elems_with(x)$

Find all elements in the current sub-tree containing x.

### flatten (filter=None)

Flatten the tree by returning a depth-first search over the tree's leaves.

### get\_index\_data(index)

Get info about the specified range in the tree.

## Returns

A dictionary with the following items:

- *elem*: The element in which index resides.
- *index*: Copy of the index parameter
- offset: The offset of index into 'elem'.

## get\_parent\_elem (child)

Searches the current sub-tree for and returns the parent of the child element.

### insert (offset, text, preferred\_parent=None)

Insert the given text at the specified offset of this string-tree's string (Unicode) representation.

## insert\_between (left, right, text)

Insert the given text between the two parameter StringElems.

### isleaf()

Whether or not this instance is a leaf node in the StringElem tree.

A node is a leaf node if it is a StringElem (not a sub-class) and contains only sub-elements of type str or unicode.

## Return type bool

### iter\_depth\_first (filter=None)

Iterate through the nodes in the tree in dept-first order.

## map (f, filter=None)

Apply f to all nodes for which filter returned True (optional).

## classmethod parse(pstr)

A parser method to extract placeables from a string based on a regular expression. Use this function as the <code>@parse()</code> method of a placeable class.

## print\_tree (indent=0, verbose=False)

Print the tree from the current instance's point in an indented manner.

#### prune()

Remove unnecessary nodes to make the tree optimal.

## remove\_type (ptype)

Replace nodes with type ptype with base StringElems, containing the same sub-elements. This is only applicable to elements below the element tree root node.

## translate()

Transform the sub-tree according to some class-specific needs. This method should be either overridden in implementing sub-classes or dynamically replaced by specific applications.

**Returns** The transformed Unicode string representing the sub-tree.

Placeable handling XML tags.

## apply\_to\_strings(f)

Apply f to all actual strings in the tree.

**Parameters f** – Must take one (str or unicode) argument and return a string or unicode.

# copy()

Returns a copy of the sub-tree. This should be overridden in sub-classes with more data.

**Note:** self.renderer is **not** copied.

### delete\_range (start\_index, end\_index)

Delete the text in the range given by the string-indexes start\_index and end\_index.

Partial nodes will only be removed if they are editable.

**Returns** A StringElem representing the removed sub-string, the parent node from which it was deleted as well as the offset at which it was deleted from. None is returned for the

parent value if the root was deleted. If the parent and offset values are not None, parent. insert (offset, deleted) effectively undoes the delete.

## depth\_first (filter=None)

Returns a list of the nodes in the tree in depth-first order.

## elem\_at\_offset (offset)

Get the StringElem in the tree that contains the string rendered at the given offset.

#### elem offset (elem)

Find the offset of elem in the current tree.

This cannot be reliably used if self.renderer is used and even less so if the rendering function renders the string differently upon different calls. In Virtual the StringElemGUI.index() method is used as replacement for this one.

**Returns** The string index where element e starts, or -1 if e was not found.

## encode (encoding='ascii')

More unicode class emulation.

#### find(x)

Find sub-string x in this string tree and return the position at which it starts.

## $find_elems_with(x)$

Find all elements in the current sub-tree containing x.

#### **flatten** (filter=None)

Flatten the tree by returning a depth-first search over the tree's leaves.

## get\_index\_data(index)

Get info about the specified range in the tree.

### Returns

A dictionary with the following items:

- *elem*: The element in which index resides.
- index: Copy of the index parameter
- offset: The offset of index into 'elem'.

# get\_parent\_elem(child)

Searches the current sub-tree for and returns the parent of the child element.

## insert (offset, text, preferred\_parent=None)

Insert the given text at the specified offset of this string-tree's string (Unicode) representation.

### insert between (left, right, text)

Insert the given text between the two parameter StringElems.

# isleaf()

Whether or not this instance is a leaf node in the StringElem tree.

A node is a leaf node if it is a StringElem (not a sub-class) and contains only sub-elements of type str or unicode.

### Return type bool

## iter\_depth\_first (filter=None)

Iterate through the nodes in the tree in dept-first order.

## map (f, filter=None)

Apply f to all nodes for which filter returned True (optional).

## classmethod parse(pstr)

A parser method to extract placeables from a string based on a regular expression. Use this function as the <code>@parse()</code> method of a placeable class.

## print\_tree (indent=0, verbose=False)

Print the tree from the current instance's point in an indented manner.

#### prune()

Remove unnecessary nodes to make the tree optimal.

## remove\_type (ptype)

Replace nodes with type ptype with base StringElems, containing the same sub-elements. This is only applicable to elements below the element tree root node.

## translate()

Transform the sub-tree according to some class-specific needs. This method should be either overridden in implementing sub-classes or dynamically replaced by specific applications.

**Returns** The transformed Unicode string representing the sub-tree.

### interfaces

This file contains abstract (semantic) interfaces for placeable implementations.

Base class for all placeables.

```
apply_to_strings(f)
```

Apply f to all actual strings in the tree.

**Parameters f** – Must take one (str or unicode) argument and return a string or unicode.

## copy()

Returns a copy of the sub-tree. This should be overridden in sub-classes with more data.

Note: self.renderer is not copied.

```
delete range (start index, end index)
```

Delete the text in the range given by the string-indexes start\_index and end\_index.

Partial nodes will only be removed if they are editable.

**Returns** A StringElem representing the removed sub-string, the parent node from which it was deleted as well as the offset at which it was deleted from. None is returned for the parent value if the root was deleted. If the parent and offset values are not None, parent. insert (offset, deleted) effectively undoes the delete.

# depth\_first (filter=None)

Returns a list of the nodes in the tree in depth-first order.

### elem\_at\_offset (offset)

Get the StringElem in the tree that contains the string rendered at the given offset.

## elem\_offset (elem)

Find the offset of elem in the current tree.

This cannot be reliably used if self.renderer is used and even less so if the rendering function renders the string differently upon different calls. In Virtual the StringElemGUI.index() method is used as replacement for this one.

**Returns** The string index where element e starts, or -1 if e was not found.

## encode (encoding='ascii')

More unicode class emulation.

#### find(x)

Find sub-string x in this string tree and return the position at which it starts.

### $find_elems_with(x)$

Find all elements in the current sub-tree containing x.

## flatten (filter=None)

Flatten the tree by returning a depth-first search over the tree's leaves.

### get\_index\_data(index)

Get info about the specified range in the tree.

#### Returns

A dictionary with the following items:

- elem: The element in which index resides.
- index: Copy of the index parameter
- offset: The offset of index into 'elem'.

# get\_parent\_elem(child)

Searches the current sub-tree for and returns the parent of the child element.

# insert (offset, text, preferred\_parent=None)

Insert the given text at the specified offset of this string-tree's string (Unicode) representation.

## insert\_between (left, right, text)

Insert the given text between the two parameter  ${\tt StringElems}.$ 

## isleaf()

Whether or not this instance is a leaf node in the StringElem tree.

A node is a leaf node if it is a StringElem (not a sub-class) and contains only sub-elements of type str or unicode.

# Return type bool

## iter\_depth\_first (filter=None)

Iterate through the nodes in the tree in dept-first order.

### map(f, filter=None)

Apply f to all nodes for which filter returned True (optional).

## print\_tree (indent=0, verbose=False)

Print the tree from the current instance's point in an indented manner.

### prune()

Remove unnecessary nodes to make the tree optimal.

## remove\_type (ptype)

Replace nodes with type ptype with base StringElems, containing the same sub-elements. This is only applicable to elements below the element tree root node.

#### translate()

Transform the sub-tree according to some class-specific needs. This method should be either overridden in implementing sub-classes or dynamically replaced by specific applications.

**Returns** The transformed Unicode string representing the sub-tree.

```
class translate.storage.placeables.interfaces.InvisiblePlaceable (sub=None, id=None, id=None, rid=None, xid=None, xid=None, **kwargs)
```

# $apply\_to\_strings(f)$

Apply f to all actual strings in the tree.

**Parameters f** – Must take one (str or unicode) argument and return a string or unicode.

## copy()

Returns a copy of the sub-tree. This should be overridden in sub-classes with more data.

Note: self.renderer is not copied.

#### **delete range** (*start index*, *end index*)

Delete the text in the range given by the string-indexes start\_index and end\_index.

Partial nodes will only be removed if they are editable.

**Returns** A StringElem representing the removed sub-string, the parent node from which it was deleted as well as the offset at which it was deleted from. None is returned for the parent value if the root was deleted. If the parent and offset values are not None, parent. insert (offset, deleted) effectively undoes the delete.

# depth\_first (filter=None)

Returns a list of the nodes in the tree in depth-first order.

#### elem at offset(offset)

Get the StringElem in the tree that contains the string rendered at the given offset.

## elem\_offset (elem)

Find the offset of elem in the current tree.

This cannot be reliably used if self.renderer is used and even less so if the rendering function renders the string differently upon different calls. In Virtual the StringElemGUI.index() method is used as replacement for this one.

**Returns** The string index where element e starts, or -1 if e was not found.

## encode (encoding='ascii')

More unicode class emulation.

# find(x)

Find sub-string x in this string tree and return the position at which it starts.

## $find_elems_with(x)$

Find all elements in the current sub-tree containing x.

## flatten (filter=None)

Flatten the tree by returning a depth-first search over the tree's leaves.

## get\_index\_data(index)

Get info about the specified range in the tree.

### Returns

A dictionary with the following items:

- elem: The element in which index resides.
- index: Copy of the index parameter
- offset: The offset of index into 'elem'.

#### get parent elem (child)

Searches the current sub-tree for and returns the parent of the child element.

```
insert (offset, text, preferred_parent=None)
```

Insert the given text at the specified offset of this string-tree's string (Unicode) representation.

### insert\_between (left, right, text)

Insert the given text between the two parameter StringElems.

### isleaf()

Whether or not this instance is a leaf node in the StringElem tree.

A node is a leaf node if it is a StringElem (not a sub-class) and contains only sub-elements of type str or unicode.

## Return type bool

## iter\_depth\_first (filter=None)

Iterate through the nodes in the tree in dept-first order.

### map(f, filter=None)

Apply f to all nodes for which filter returned True (optional).

## print\_tree (indent=0, verbose=False)

Print the tree from the current instance's point in an indented manner.

## prune()

Remove unnecessary nodes to make the tree optimal.

## remove\_type (ptype)

Replace nodes with type ptype with base StringElems, containing the same sub-elements. This is only applicable to elements below the element tree root node.

# translate()

Transform the sub-tree according to some class-specific needs. This method should be either overridden in implementing sub-classes or dynamically replaced by specific applications.

**Returns** The transformed Unicode string representing the sub-tree.

```
class translate.storage.placeables.interfaces.MaskingPlaceable (sub=None, id=None, rid=None, xid=None, x
```

## $apply_to_strings(f)$

Apply f to all actual strings in the tree.

**Parameters f** – Must take one (str or unicode) argument and return a string or unicode.

### copy()

Returns a copy of the sub-tree. This should be overridden in sub-classes with more data.

Note: self.renderer is not copied.

## delete\_range (start\_index, end\_index)

Delete the text in the range given by the string-indexes start\_index and end\_index.

Partial nodes will only be removed if they are editable.

**Returns** A StringElem representing the removed sub-string, the parent node from which it was deleted as well as the offset at which it was deleted from. None is returned for the parent value if the root was deleted. If the parent and offset values are not None, parent. insert (offset, deleted) effectively undoes the delete.

## depth\_first (filter=None)

Returns a list of the nodes in the tree in depth-first order.

### elem\_at\_offset (offset)

Get the StringElem in the tree that contains the string rendered at the given offset.

## elem\_offset (elem)

Find the offset of elem in the current tree.

This cannot be reliably used if self.renderer is used and even less so if the rendering function renders the string differently upon different calls. In Virtual the StringElemGUI.index() method is used as replacement for this one.

**Returns** The string index where element e starts, or -1 if e was not found.

#### encode (encoding='ascii')

More unicode class emulation.

# find(x)

Find sub-string x in this string tree and return the position at which it starts.

## $find_elems_with(x)$

Find all elements in the current sub-tree containing x.

### flatten (filter=None)

Flatten the tree by returning a depth-first search over the tree's leaves.

## get\_index\_data(index)

Get info about the specified range in the tree.

#### Returns

A dictionary with the following items:

- elem: The element in which index resides.
- *index*: Copy of the index parameter
- offset: The offset of index into 'elem'.

# get\_parent\_elem (child)

Searches the current sub-tree for and returns the parent of the child element.

### insert (offset, text, preferred\_parent=None)

Insert the given text at the specified offset of this string-tree's string (Unicode) representation.

## insert\_between (left, right, text)

Insert the given text between the two parameter StringElems.

#### isleaf()

Whether or not this instance is a leaf node in the StringElem tree.

A node is a leaf node if it is a StringElem (not a sub-class) and contains only sub-elements of type str or unicode.

# Return type bool

## iter\_depth\_first (filter=None)

Iterate through the nodes in the tree in dept-first order.

```
map (f, filter=None)
```

Apply f to all nodes for which filter returned True (optional).

### print\_tree (indent=0, verbose=False)

Print the tree from the current instance's point in an indented manner.

### prune()

Remove unnecessary nodes to make the tree optimal.

### remove\_type (ptype)

Replace nodes with type ptype with base StringElems, containing the same sub-elements. This is only applicable to elements below the element tree root node.

### translate()

Transform the sub-tree according to some class-specific needs. This method should be either overridden in implementing sub-classes or dynamically replaced by specific applications.

**Returns** The transformed Unicode string representing the sub-tree.

## apply\_to\_strings(f)

Apply f to all actual strings in the tree.

**Parameters f** – Must take one (str or unicode) argument and return a string or unicode.

## copy()

Returns a copy of the sub-tree. This should be overridden in sub-classes with more data.

Note: self.renderer is not copied.

## delete\_range (start\_index, end\_index)

Delete the text in the range given by the string-indexes start\_index and end\_index.

Partial nodes will only be removed if they are editable.

**Returns** A StringElem representing the removed sub-string, the parent node from which it was deleted as well as the offset at which it was deleted from. None is returned for the parent value if the root was deleted. If the parent and offset values are not None, parent. insert (offset, deleted) effectively undoes the delete.

## depth\_first (filter=None)

Returns a list of the nodes in the tree in depth-first order.

## elem\_at\_offset (offset)

Get the StringElem in the tree that contains the string rendered at the given offset.

#### elem offset (elem)

Find the offset of elem in the current tree.

This cannot be reliably used if self.renderer is used and even less so if the rendering function renders the string differently upon different calls. In Virtual the StringElemGUI.index() method is used as replacement for this one.

**Returns** The string index where element e starts, or -1 if e was not found.

### encode (encoding='ascii')

More unicode class emulation.

### find(x)

Find sub-string x in this string tree and return the position at which it starts.

## $find_elems_with(x)$

Find all elements in the current sub-tree containing x.

## flatten (filter=None)

Flatten the tree by returning a depth-first search over the tree's leaves.

### get index data(index)

Get info about the specified range in the tree.

#### Returns

A dictionary with the following items:

- elem: The element in which index resides.
- *index*: Copy of the index parameter
- offset: The offset of index into 'elem'.

## get\_parent\_elem(child)

Searches the current sub-tree for and returns the parent of the child element.

## insert (offset, text, preferred\_parent=None)

Insert the given text at the specified offset of this string-tree's string (Unicode) representation.

## insert\_between (left, right, text)

Insert the given text between the two parameter StringElems.

# isleaf()

Whether or not this instance is a leaf node in the StringElem tree.

A node is a leaf node if it is a StringElem (not a sub-class) and contains only sub-elements of type str or unicode.

### Return type bool

### iter\_depth\_first(filter=None)

Iterate through the nodes in the tree in dept-first order.

# map(f, filter=None)

Apply f to all nodes for which filter returned True (optional).

### print\_tree (indent=0, verbose=False)

Print the tree from the current instance's point in an indented manner.

## prune()

Remove unnecessary nodes to make the tree optimal.

## remove\_type (ptype)

Replace nodes with type ptype with base StringElems, containing the same sub-elements. This is only applicable to elements below the element tree root node.

### translate()

Transform the sub-tree according to some class-specific needs. This method should be either overridden in implementing sub-classes or dynamically replaced by specific applications.

**Returns** The transformed Unicode string representing the sub-tree.

```
class translate.storage.placeables.interfaces.SubflowPlaceable (sub=None, id=None, rid=None, xid=None, x
```

### apply\_to\_strings(f)

Apply f to all actual strings in the tree.

**Parameters f** – Must take one (str or unicode) argument and return a string or unicode.

## copy()

Returns a copy of the sub-tree. This should be overridden in sub-classes with more data.

Note: self.renderer is not copied.

## delete\_range (start\_index, end\_index)

Delete the text in the range given by the string-indexes start\_index and end\_index.

Partial nodes will only be removed if they are editable.

Returns A StringElem representing the removed sub-string, the parent node from which it was deleted as well as the offset at which it was deleted from. None is returned for the parent value if the root was deleted. If the parent and offset values are not None, parent. insert (offset, deleted) effectively undoes the delete.

### depth\_first (filter=None)

Returns a list of the nodes in the tree in depth-first order.

# elem\_at\_offset (offset)

Get the StringElem in the tree that contains the string rendered at the given offset.

#### elem offset(elem)

Find the offset of elem in the current tree.

This cannot be reliably used if self.renderer is used and even less so if the rendering function renders the string differently upon different calls. In Virtual the StringElemGUI.index() method is used as replacement for this one.

**Returns** The string index where element e starts, or -1 if e was not found.

## encode (encoding='ascii')

More unicode class emulation.

#### find(x)

Find sub-string x in this string tree and return the position at which it starts.

## $find_elems_with(x)$

Find all elements in the current sub-tree containing x.

### flatten (filter=None)

Flatten the tree by returning a depth-first search over the tree's leaves.

## get\_index\_data(index)

Get info about the specified range in the tree.

### **Returns**

A dictionary with the following items:

- elem: The element in which index resides.
- *index*: Copy of the index parameter
- offset: The offset of index into 'elem'.

### get\_parent\_elem(child)

Searches the current sub-tree for and returns the parent of the child element.

## insert (offset, text, preferred\_parent=None)

Insert the given text at the specified offset of this string-tree's string (Unicode) representation.

#### insert between (left, right, text)

Insert the given text between the two parameter StringElems.

# isleaf()

Whether or not this instance is a leaf node in the StringElem tree.

A node is a leaf node if it is a StringElem (not a sub-class) and contains only sub-elements of type str or unicode.

# Return type bool

## iter\_depth\_first(filter=None)

Iterate through the nodes in the tree in dept-first order.

# map (f, filter=None)

Apply f to all nodes for which filter returned True (optional).

### print\_tree (indent=0, verbose=False)

Print the tree from the current instance's point in an indented manner.

### prune()

Remove unnecessary nodes to make the tree optimal.

## remove\_type (ptype)

Replace nodes with type ptype with base StringElems, containing the same sub-elements. This is only applicable to elements below the element tree root node.

### translate()

Transform the sub-tree according to some class-specific needs. This method should be either overridden in implementing sub-classes or dynamically replaced by specific applications.

**Returns** The transformed Unicode string representing the sub-tree.

## lisa

### parse

Contains the parse function that parses normal strings into StringElem- based "rich" string element trees.

```
translate.storage.placeables.parse.parse(tree, parse_funcs)
```

Parse placeables from the given string or sub-tree by using the parsing functions provided.

The output of this function is **heavily** dependent on the order of the parsing functions. This is because of the algorithm used.

An over-simplification of the algorithm: the leaves in the StringElem tree are expanded to the output of the first parsing function in parse\_funcs. The next level of recursion is then started on the new set of leaves with the used parsing function removed from parse\_funcs.

**Parameters tree** (unicode/StringElem) – The string or string element sub-tree to parse.

#### strelem

Contains the base *StringElem* class that represents a node in a parsed rich-string tree. It is the base class of all placeables.

```
exception translate.storage.placeables.strelem.ElementNotFoundError
```

This class represents a sub-tree of a string parsed into a rich structure. It is also the base class of all placeables.

```
apply_to_strings(f)
```

Apply f to all actual strings in the tree.

**Parameters f** – Must take one (str or unicode) argument and return a string or unicode.

### copy()

Returns a copy of the sub-tree. This should be overridden in sub-classes with more data.

```
Note: self.renderer is not copied.
```

```
delete_range (start_index, end_index)
```

Delete the text in the range given by the string-indexes start\_index and end\_index.

Partial nodes will only be removed if they are editable.

Returns A StringElem representing the removed sub-string, the parent node from which it was deleted as well as the offset at which it was deleted from. None is returned for the parent value if the root was deleted. If the parent and offset values are not None, parent. insert (offset, deleted) effectively undoes the delete.

# depth\_first (filter=None)

Returns a list of the nodes in the tree in depth-first order.

#### elem\_at\_offset (offset)

Get the StringElem in the tree that contains the string rendered at the given offset.

# ${\tt elem\_offset}\;(elem)$

Find the offset of elem in the current tree.

This cannot be reliably used if self.renderer is used and even less so if the rendering function renders the string differently upon different calls. In Virtual the StringElemGUI.index() method is used as replacement for this one.

**Returns** The string index where element e starts, or -1 if e was not found.

### encode (encoding='ascii')

More unicode class emulation.

#### find(x)

Find sub-string x in this string tree and return the position at which it starts.

## $find_elems_with(x)$

Find all elements in the current sub-tree containing x.

### flatten (filter=None)

Flatten the tree by returning a depth-first search over the tree's leaves.

## get\_index\_data(index)

Get info about the specified range in the tree.

#### Returns

A dictionary with the following items:

- elem: The element in which index resides.
- *index*: Copy of the index parameter
- offset: The offset of index into 'elem'.

## get\_parent\_elem(child)

Searches the current sub-tree for and returns the parent of the child element.

#### has content = True

Whether this string can have sub-elements.

## insert (offset, text, preferred\_parent=None)

Insert the given text at the specified offset of this string-tree's string (Unicode) representation.

## insert\_between (left, right, text)

Insert the given text between the two parameter StringElems.

### iseditable = True

Whether this string should be changable by the user. Not used at the moment.

### isfragile = False

Whether this element should be deleted in its entirety when partially deleted. Only checked when iseditable = False

## isleaf()

Whether or not this instance is a leaf node in the StringElem tree.

A node is a leaf node if it is a StringElem (not a sub-class) and contains only sub-elements of type str or unicode.

### Return type bool

# istranslatable = True

Whether this string is translatable into other languages.

## isvisible = True

Whether this string should be visible to the user. Not used at the moment.

### iter\_depth\_first (filter=None)

Iterate through the nodes in the tree in dept-first order.

## map (f, filter=None)

Apply f to all nodes for which filter returned True (optional).

### classmethod parse(pstr)

Parse an instance of this class from the start of the given string. This method should be implemented by any sub-class that wants to parseable by translate.storage.placeables.parse.

**Parameters** pstr (unicode) – The string to parse into an instance of this class.

**Returns** An instance of the current class, or None if the string not parseable by this class.

## print\_tree (indent=0, verbose=False)

Print the tree from the current instance's point in an indented manner.

### prune()

Remove unnecessary nodes to make the tree optimal.

# remove\_type (ptype)

Replace nodes with type ptype with base StringElems, containing the same sub-elements. This is only applicable to elements below the element tree root node.

#### renderer = None

An optional function that returns the Unicode representation of the string.

#### sub = []

The sub-elements that make up this this string.

#### translate()

Transform the sub-tree according to some class-specific needs. This method should be either overridden in implementing sub-classes or dynamically replaced by specific applications.

**Returns** The transformed Unicode string representing the sub-tree.

## terminology

Contains the placeable that represents a terminology term.

Terminology distinguished from the rest of a string by being a placeable.

### apply\_to\_strings(f)

Apply f to all actual strings in the tree.

**Parameters f** – Must take one (str or unicode) argument and return a string or unicode.

#### copy()

Returns a copy of the sub-tree. This should be overridden in sub-classes with more data.

Note: self.renderer is not copied.

## delete\_range (start\_index, end\_index)

Delete the text in the range given by the string-indexes start\_index and end\_index.

Partial nodes will only be removed if they are editable.

**Returns** A StringElem representing the removed sub-string, the parent node from which it was deleted as well as the offset at which it was deleted from. None is returned for the parent value if the root was deleted. If the parent and offset values are not None, parent. insert (offset, deleted) effectively undoes the delete.

# depth\_first (filter=None)

Returns a list of the nodes in the tree in depth-first order.

### elem at offset(offset)

Get the StringElem in the tree that contains the string rendered at the given offset.

## elem\_offset (elem)

Find the offset of elem in the current tree.

This cannot be reliably used if self.renderer is used and even less so if the rendering function renders the string differently upon different calls. In Virtual the StringElemGUI.index() method is used as replacement for this one.

**Returns** The string index where element e starts, or -1 if e was not found.

## encode (encoding='ascii')

More unicode class emulation.

#### find(x)

Find sub-string x in this string tree and return the position at which it starts.

### $find_elems_with(x)$

Find all elements in the current sub-tree containing x.

### flatten (filter=None)

Flatten the tree by returning a depth-first search over the tree's leaves.

## get\_index\_data(index)

Get info about the specified range in the tree.

#### **Returns**

A dictionary with the following items:

- elem: The element in which index resides.
- index: Copy of the index parameter
- offset: The offset of index into 'elem'.

## get\_parent\_elem (child)

Searches the current sub-tree for and returns the parent of the child element.

## insert (offset, text, preferred\_parent=None)

Insert the given text at the specified offset of this string-tree's string (Unicode) representation.

## insert\_between (left, right, text)

Insert the given text between the two parameter StringElems.

## isleaf()

Whether or not this instance is a leaf node in the StringElem tree.

A node is a leaf node if it is a StringElem (not a sub-class) and contains only sub-elements of type str or unicode.

# Return type bool

## iter\_depth\_first(filter=None)

Iterate through the nodes in the tree in dept-first order.

### map (f, filter=None)

Apply f to all nodes for which filter returned True (optional).

## matchers = []

A list of matcher objects to use to identify terminology.

## classmethod parse(pstr)

Parse an instance of this class from the start of the given string. This method should be implemented by any sub-class that wants to parseable by translate.storage.placeables.parse.

**Parameters** pstr (unicode) – The string to parse into an instance of this class.

Returns An instance of the current class, or None if the string not parseable by this class.

# print\_tree (indent=0, verbose=False)

Print the tree from the current instance's point in an indented manner.

### prune()

Remove unnecessary nodes to make the tree optimal.

# remove\_type (ptype)

Replace nodes with type ptype with base StringElems, containing the same sub-elements. This is only applicable to elements below the element tree root node.

## translate()

Transform the sub-tree according to some class-specific needs. This method should be either overridden in implementing sub-classes or dynamically replaced by specific applications.

**Returns** The transformed Unicode string representing the sub-tree.

## translations = []

The available translations for this placeable.

#### xliff

Contains XLIFF-specific placeables.

```
class translate.storage.placeables.xliff.Bpt (sub=None, id=None, rid=None, xid=None, xid=No
```

# ${\tt apply\_to\_strings}\,(f)$

Apply f to all actual strings in the tree.

**Parameters f** – Must take one (str or unicode) argument and return a string or unicode.

## copy()

Returns a copy of the sub-tree. This should be overridden in sub-classes with more data.

Note: self.renderer is not copied.

## delete\_range (start\_index, end\_index)

Delete the text in the range given by the string-indexes start\_index and end\_index.

Partial nodes will only be removed if they are editable.

Returns A StringElem representing the removed sub-string, the parent node from which it was deleted as well as the offset at which it was deleted from. None is returned for the parent value if the root was deleted. If the parent and offset values are not None, parent. insert (offset, deleted) effectively undoes the delete.

### depth\_first (filter=None)

Returns a list of the nodes in the tree in depth-first order.

# elem\_at\_offset (offset)

Get the StringElem in the tree that contains the string rendered at the given offset.

#### elem offset (elem)

Find the offset of elem in the current tree.

This cannot be reliably used if self.renderer is used and even less so if the rendering function renders the string differently upon different calls. In Virtual the StringElemGUI.index() method is used as replacement for this one.

**Returns** The string index where element e starts, or -1 if e was not found.

### encode (encoding='ascii')

More unicode class emulation.

## find(x)

Find sub-string x in this string tree and return the position at which it starts.

## $find_elems_with(x)$

Find all elements in the current sub-tree containing x.

### flatten (filter=None)

Flatten the tree by returning a depth-first search over the tree's leaves.

### get index data(index)

Get info about the specified range in the tree.

#### Returns

A dictionary with the following items:

- elem: The element in which index resides.
- *index*: Copy of the index parameter
- offset: The offset of index into 'elem'.

## get\_parent\_elem(child)

Searches the current sub-tree for and returns the parent of the child element.

# insert (offset, text, preferred\_parent=None)

Insert the given text at the specified offset of this string-tree's string (Unicode) representation.

## insert\_between (left, right, text)

Insert the given text between the two parameter StringElems.

# isleaf()

Whether or not this instance is a leaf node in the StringElem tree.

A node is a leaf node if it is a StringElem (not a sub-class) and contains only sub-elements of type str or unicode.

### Return type bool

### iter\_depth\_first(filter=None)

Iterate through the nodes in the tree in dept-first order.

# map(f, filter=None)

Apply f to all nodes for which filter returned True (optional).

### print\_tree (indent=0, verbose=False)

Print the tree from the current instance's point in an indented manner.

## prune()

Remove unnecessary nodes to make the tree optimal.

# remove\_type (ptype)

Replace nodes with type ptype with base StringElems, containing the same sub-elements. This is only applicable to elements below the element tree root node.

### translate()

Transform the sub-tree according to some class-specific needs. This method should be either overridden in implementing sub-classes or dynamically replaced by specific applications.

**Returns** The transformed Unicode string representing the sub-tree.

class translate.storage.placeables.xliff.Ept (sub=None, id=None, rid=None, xid=None, xid=No

## apply\_to\_strings(f)

Apply f to all actual strings in the tree.

**Parameters f** – Must take one (str or unicode) argument and return a string or unicode.

#### copy()

Returns a copy of the sub-tree. This should be overridden in sub-classes with more data.

**Note:** self.renderer is **not** copied.

#### **delete range** (*start index*, *end index*)

Delete the text in the range given by the string-indexes start\_index and end\_index.

Partial nodes will only be removed if they are editable.

**Returns** A StringElem representing the removed sub-string, the parent node from which it was deleted as well as the offset at which it was deleted from. None is returned for the parent value if the root was deleted. If the parent and offset values are not None, parent. insert (offset, deleted) effectively undoes the delete.

# depth\_first (filter=None)

Returns a list of the nodes in the tree in depth-first order.

#### elem at offset(offset)

Get the StringElem in the tree that contains the string rendered at the given offset.

## $elem\_offset (elem)$

Find the offset of elem in the current tree.

This cannot be reliably used if self.renderer is used and even less so if the rendering function renders the string differently upon different calls. In Virtual the StringElemGUI.index() method is used as replacement for this one.

**Returns** The string index where element e starts, or -1 if e was not found.

## encode (encoding='ascii')

More unicode class emulation.

# find(x)

Find sub-string x in this string tree and return the position at which it starts.

### $find_elems_with(x)$

Find all elements in the current sub-tree containing x.

## flatten (filter=None)

Flatten the tree by returning a depth-first search over the tree's leaves.

## get\_index\_data(index)

Get info about the specified range in the tree.

### Returns

A dictionary with the following items:

- elem: The element in which index resides.
- index: Copy of the index parameter
- offset: The offset of index into 'elem'.

#### get parent elem (child)

Searches the current sub-tree for and returns the parent of the child element.

# insert (offset, text, preferred\_parent=None)

Insert the given text at the specified offset of this string-tree's string (Unicode) representation.

### insert\_between (left, right, text)

Insert the given text between the two parameter StringElems.

### isleaf()

Whether or not this instance is a leaf node in the StringElem tree.

A node is a leaf node if it is a StringElem (not a sub-class) and contains only sub-elements of type str or unicode.

# Return type bool

## iter\_depth\_first (filter=None)

Iterate through the nodes in the tree in dept-first order.

### map(f, filter=None)

Apply f to all nodes for which filter returned True (optional).

## print\_tree (indent=0, verbose=False)

Print the tree from the current instance's point in an indented manner.

# prune()

Remove unnecessary nodes to make the tree optimal.

## remove\_type (ptype)

Replace nodes with type ptype with base StringElems, containing the same sub-elements. This is only applicable to elements below the element tree root node.

# translate()

Transform the sub-tree according to some class-specific needs. This method should be either overridden in implementing sub-classes or dynamically replaced by specific applications.

**Returns** The transformed Unicode string representing the sub-tree.

**class** translate.storage.placeables.xliff.**X**(*id=None*, *xid=None*, \*\*kwargs)

# $apply_to_strings(f)$

Apply f to all actual strings in the tree.

**Parameters f** – Must take one (str or unicode) argument and return a string or unicode.

## copy()

Returns a copy of the sub-tree. This should be overridden in sub-classes with more data.

Note: self.renderer is not copied.

### **delete range** (*start index*, *end index*)

Delete the text in the range given by the string-indexes start\_index and end\_index.

Partial nodes will only be removed if they are editable.

**Returns** A StringElem representing the removed sub-string, the parent node from which it was deleted as well as the offset at which it was deleted from. None is returned for the parent value if the root was deleted. If the parent and offset values are not None, parent. insert (offset, deleted) effectively undoes the delete.

## depth\_first (filter=None)

Returns a list of the nodes in the tree in depth-first order.

### elem\_at\_offset (offset)

Get the StringElem in the tree that contains the string rendered at the given offset.

#### elem offset (elem)

Find the offset of elem in the current tree.

This cannot be reliably used if self.renderer is used and even less so if the rendering function renders the string differently upon different calls. In Virtual the StringElemGUI.index() method is used as replacement for this one.

**Returns** The string index where element e starts, or -1 if e was not found.

### encode (encoding='ascii')

More unicode class emulation.

#### find(x)

Find sub-string x in this string tree and return the position at which it starts.

### $find_elems_with(x)$

Find all elements in the current sub-tree containing x.

## flatten (filter=None)

Flatten the tree by returning a depth-first search over the tree's leaves.

## get\_index\_data(index)

Get info about the specified range in the tree.

# Returns

A dictionary with the following items:

- *elem*: The element in which index resides.
- *index*: Copy of the index parameter
- offset: The offset of index into 'elem'.

# ${\tt get\_parent\_elem}\,(child)$

Searches the current sub-tree for and returns the parent of the child element.

## insert (offset, text, preferred\_parent=None)

Insert the given text at the specified offset of this string-tree's string (Unicode) representation.

### insert\_between (left, right, text)

Insert the given text between the two parameter StringElems.

# isleaf()

Whether or not this instance is a leaf node in the StringElem tree.

A node is a leaf node if it is a StringElem (not a sub-class) and contains only sub-elements of type str or unicode.

### Return type bool

## iter\_depth\_first (filter=None)

Iterate through the nodes in the tree in dept-first order.

## map (f, filter=None)

Apply f to all nodes for which filter returned True (optional).

# print\_tree (indent=0, verbose=False)

Print the tree from the current instance's point in an indented manner.

### prune()

Remove unnecessary nodes to make the tree optimal.

# remove\_type (ptype)

Replace nodes with type ptype with base StringElems, containing the same sub-elements. This is only applicable to elements below the element tree root node.

## translate()

Transform the sub-tree according to some class-specific needs. This method should be either overridden in implementing sub-classes or dynamically replaced by specific applications.

**Returns** The transformed Unicode string representing the sub-tree.

class translate.storage.placeables.xliff.Bx (id=None, xid=None, \*\*kwargs)

### apply\_to\_strings(f)

Apply f to all actual strings in the tree.

**Parameters f** – Must take one (str or unicode) argument and return a string or unicode.

## copy()

Returns a copy of the sub-tree. This should be overridden in sub-classes with more data.

Note: self.renderer is not copied.

## delete\_range (start\_index, end\_index)

Delete the text in the range given by the string-indexes start\_index and end\_index.

Partial nodes will only be removed if they are editable.

Returns A StringElem representing the removed sub-string, the parent node from which it was deleted as well as the offset at which it was deleted from. None is returned for the parent value if the root was deleted. If the parent and offset values are not None, parent. insert (offset, deleted) effectively undoes the delete.

### depth\_first (filter=None)

Returns a list of the nodes in the tree in depth-first order.

# elem\_at\_offset (offset)

Get the StringElem in the tree that contains the string rendered at the given offset.

#### elem offset(elem)

Find the offset of elem in the current tree.

This cannot be reliably used if self.renderer is used and even less so if the rendering function renders the string differently upon different calls. In Virtual the StringElemGUI.index() method is used as replacement for this one.

**Returns** The string index where element e starts, or -1 if e was not found.

### encode (encoding='ascii')

More unicode class emulation.

#### find(x)

Find sub-string x in this string tree and return the position at which it starts.

## $find_elems_with(x)$

Find all elements in the current sub-tree containing x.

#### flatten (filter=None)

Flatten the tree by returning a depth-first search over the tree's leaves.

## get\_index\_data(index)

Get info about the specified range in the tree.

#### Returns

A dictionary with the following items:

- elem: The element in which index resides.
- *index*: Copy of the index parameter
- offset: The offset of index into 'elem'.

## get\_parent\_elem(child)

Searches the current sub-tree for and returns the parent of the child element.

## insert (offset, text, preferred\_parent=None)

Insert the given text at the specified offset of this string-tree's string (Unicode) representation.

## insert\_between (left, right, text)

Insert the given text between the two parameter StringElems.

# isleaf()

Whether or not this instance is a leaf node in the StringElem tree.

A node is a leaf node if it is a StringElem (not a sub-class) and contains only sub-elements of type str or unicode.

## Return type bool

### iter\_depth\_first (filter=None)

Iterate through the nodes in the tree in dept-first order.

## map(f, filter=None)

Apply f to all nodes for which filter returned True (optional).

## print\_tree (indent=0, verbose=False)

Print the tree from the current instance's point in an indented manner.

### prune()

Remove unnecessary nodes to make the tree optimal.

# remove\_type (ptype)

Replace nodes with type ptype with base StringElems, containing the same sub-elements. This is only applicable to elements below the element tree root node.

### translate()

Transform the sub-tree according to some class-specific needs. This method should be either overridden in implementing sub-classes or dynamically replaced by specific applications.

**Returns** The transformed Unicode string representing the sub-tree.

**class** translate.storage.placeables.xliff.**Ex**(*id=None*, *xid=None*, \*\*kwargs)

# $apply_to_strings(f)$

Apply f to all actual strings in the tree.

**Parameters f** – Must take one (str or unicode) argument and return a string or unicode.

#### copy()

Returns a copy of the sub-tree. This should be overridden in sub-classes with more data.

Note: self.renderer is not copied.

### delete\_range (start\_index, end\_index)

Delete the text in the range given by the string-indexes start\_index and end\_index.

Partial nodes will only be removed if they are editable.

Returns A StringElem representing the removed sub-string, the parent node from which it was deleted as well as the offset at which it was deleted from. None is returned for the parent value if the root was deleted. If the parent and offset values are not None, parent. insert (offset, deleted) effectively undoes the delete.

#### depth first(filter=None)

Returns a list of the nodes in the tree in depth-first order.

### elem\_at\_offset (offset)

Get the StringElem in the tree that contains the string rendered at the given offset.

#### elem\_offset (elem)

Find the offset of elem in the current tree.

This cannot be reliably used if self.renderer is used and even less so if the rendering function renders the string differently upon different calls. In Virtual the StringElemGUI.index() method is used as replacement for this one.

**Returns** The string index where element e starts, or -1 if e was not found.

#### encode (encoding='ascii')

More unicode class emulation.

## find(x)

Find sub-string x in this string tree and return the position at which it starts.

# $find_elems_with(x)$

Find all elements in the current sub-tree containing x.

#### **flatten** (filter=None)

Flatten the tree by returning a depth-first search over the tree's leaves.

# get\_index\_data(index)

Get info about the specified range in the tree.

#### Returns

A dictionary with the following items:

- *elem*: The element in which index resides.
- index: Copy of the index parameter
- offset: The offset of index into 'elem'.

#### get\_parent\_elem(child)

Searches the current sub-tree for and returns the parent of the child element.

### insert (offset, text, preferred\_parent=None)

Insert the given text at the specified offset of this string-tree's string (Unicode) representation.

### insert\_between (left, right, text)

Insert the given text between the two parameter StringElems.

#### isleaf()

Whether or not this instance is a leaf node in the StringElem tree.

A node is a leaf node if it is a StringElem (not a sub-class) and contains only sub-elements of type str or unicode.

## Return type bool

# iter\_depth\_first (filter=None)

Iterate through the nodes in the tree in dept-first order.

#### map(f, filter=None)

Apply f to all nodes for which filter returned True (optional).

#### print\_tree (indent=0, verbose=False)

Print the tree from the current instance's point in an indented manner.

### prune()

Remove unnecessary nodes to make the tree optimal.

#### remove\_type (ptype)

Replace nodes with type ptype with base StringElems, containing the same sub-elements. This is only applicable to elements below the element tree root node.

### translate()

Transform the sub-tree according to some class-specific needs. This method should be either overridden in implementing sub-classes or dynamically replaced by specific applications.

**Returns** The transformed Unicode string representing the sub-tree.

### apply\_to\_strings(f)

Apply f to all actual strings in the tree.

**Parameters f** – Must take one (str or unicode) argument and return a string or unicode.

### copy()

Returns a copy of the sub-tree. This should be overridden in sub-classes with more data.

**Note:** self.renderer is **not** copied.

### delete\_range (start\_index, end\_index)

Delete the text in the range given by the string-indexes start\_index and end\_index.

Partial nodes will only be removed if they are editable.

Returns A StringElem representing the removed sub-string, the parent node from which it was deleted as well as the offset at which it was deleted from. None is returned for the parent value if the root was deleted. If the parent and offset values are not None, parent. insert (offset, deleted) effectively undoes the delete.

#### depth\_first (filter=None)

Returns a list of the nodes in the tree in depth-first order.

### elem\_at\_offset (offset)

Get the StringElem in the tree that contains the string rendered at the given offset.

# $elem\_offset(elem)$

Find the offset of elem in the current tree.

This cannot be reliably used if self.renderer is used and even less so if the rendering function renders the string differently upon different calls. In Virtual the StringElemGUI.index() method is used as replacement for this one.

**Returns** The string index where element e starts, or -1 if e was not found.

#### encode (encoding='ascii')

More unicode class emulation.

#### find(x)

Find sub-string x in this string tree and return the position at which it starts.

#### find elems with (x)

Find all elements in the current sub-tree containing x.

## flatten (filter=None)

Flatten the tree by returning a depth-first search over the tree's leaves.

#### get\_index\_data(index)

Get info about the specified range in the tree.

#### Returns

A dictionary with the following items:

- elem: The element in which index resides.
- index: Copy of the index parameter
- offset: The offset of index into 'elem'.

### get\_parent\_elem(child)

Searches the current sub-tree for and returns the parent of the child element.

# insert (offset, text, preferred\_parent=None)

Insert the given text at the specified offset of this string-tree's string (Unicode) representation.

## insert\_between (left, right, text)

Insert the given text between the two parameter StringElems.

#### isleaf()

Whether or not this instance is a leaf node in the StringElem tree.

A node is a leaf node if it is a StringElem (not a sub-class) and contains only sub-elements of type str or unicode.

# Return type bool

#### iter\_depth\_first (filter=None)

Iterate through the nodes in the tree in dept-first order.

### map(f, filter=None)

Apply f to all nodes for which filter returned True (optional).

# print\_tree (indent=0, verbose=False)

Print the tree from the current instance's point in an indented manner.

#### prune()

Remove unnecessary nodes to make the tree optimal.

# remove\_type (ptype)

Replace nodes with type ptype with base StringElems, containing the same sub-elements. This is only applicable to elements below the element tree root node.

#### translate()

Transform the sub-tree according to some class-specific needs. This method should be either overridden in implementing sub-classes or dynamically replaced by specific applications.

**Returns** The transformed Unicode string representing the sub-tree.

## $apply_to_strings(f)$

Apply f to all actual strings in the tree.

**Parameters f** – Must take one (str or unicode) argument and return a string or unicode.

## copy()

Returns a copy of the sub-tree. This should be overridden in sub-classes with more data.

Note: self.renderer is not copied.

### delete\_range (start\_index, end\_index)

Delete the text in the range given by the string-indexes start index and end index.

Partial nodes will only be removed if they are editable.

Returns A StringElem representing the removed sub-string, the parent node from which it was deleted as well as the offset at which it was deleted from. None is returned for the parent value if the root was deleted. If the parent and offset values are not None, parent. insert (offset, deleted) effectively undoes the delete.

#### depth\_first (filter=None)

Returns a list of the nodes in the tree in depth-first order.

## elem\_at\_offset (offset)

Get the StringElem in the tree that contains the string rendered at the given offset.

# elem\_offset (elem)

Find the offset of elem in the current tree.

This cannot be reliably used if self.renderer is used and even less so if the rendering function renders the string differently upon different calls. In Virtual the StringElemGUI.index() method is used as replacement for this one.

**Returns** The string index where element e starts, or -1 if e was not found.

# encode (encoding='ascii')

More unicode class emulation.

#### find(x)

Find sub-string x in this string tree and return the position at which it starts.

### $find_elems_with(x)$

Find all elements in the current sub-tree containing x.

## flatten (filter=None)

Flatten the tree by returning a depth-first search over the tree's leaves.

#### get\_index\_data(index)

Get info about the specified range in the tree.

#### Returns

A dictionary with the following items:

- *elem*: The element in which index resides.
- *index*: Copy of the index parameter
- offset: The offset of index into 'elem'.

### get\_parent\_elem(child)

Searches the current sub-tree for and returns the parent of the child element.

### insert (offset, text, preferred\_parent=None)

Insert the given text at the specified offset of this string-tree's string (Unicode) representation.

## insert\_between (left, right, text)

Insert the given text between the two parameter StringElems.

#### isleaf(

Whether or not this instance is a leaf node in the StringElem tree.

A node is a leaf node if it is a StringElem (not a sub-class) and contains only sub-elements of type str or unicode.

#### Return type bool

### iter\_depth\_first (filter=None)

Iterate through the nodes in the tree in dept-first order.

### map (f, filter=None)

Apply f to all nodes for which filter returned True (optional).

# print\_tree (indent=0, verbose=False)

Print the tree from the current instance's point in an indented manner.

### prune()

Remove unnecessary nodes to make the tree optimal.

### remove\_type (ptype)

Replace nodes with type ptype with base StringElems, containing the same sub-elements. This is only applicable to elements below the element tree root node.

### translate()

Transform the sub-tree according to some class-specific needs. This method should be either overridden in implementing sub-classes or dynamically replaced by specific applications.

**Returns** The transformed Unicode string representing the sub-tree.

class translate.storage.placeables.xliff.Sub(sub=None, id=None, rid=None, xid=None, xid=Non

# ${\tt apply\_to\_strings}\,(f)$

Apply f to all actual strings in the tree.

**Parameters f** – Must take one (str or unicode) argument and return a string or unicode.

### copy()

Returns a copy of the sub-tree. This should be overridden in sub-classes with more data.

Note: self.renderer is not copied.

#### delete\_range (start\_index, end\_index)

Delete the text in the range given by the string-indexes start\_index and end\_index.

Partial nodes will only be removed if they are editable.

**Returns** A StringElem representing the removed sub-string, the parent node from which it was deleted as well as the offset at which it was deleted from. None is returned for the parent value if the root was deleted. If the parent and offset values are not None, parent. insert (offset, deleted) effectively undoes the delete.

### depth\_first (filter=None)

Returns a list of the nodes in the tree in depth-first order.

#### elem\_at\_offset (offset)

Get the StringElem in the tree that contains the string rendered at the given offset.

### elem\_offset (elem)

Find the offset of elem in the current tree.

This cannot be reliably used if self.renderer is used and even less so if the rendering function renders the string differently upon different calls. In Virtual the StringElemGUI.index() method is used as replacement for this one.

**Returns** The string index where element e starts, or -1 if e was not found.

#### encode (encoding='ascii')

More unicode class emulation.

# find(x)

Find sub-string x in this string tree and return the position at which it starts.

### $find_elems_with(x)$

Find all elements in the current sub-tree containing x.

#### flatten (filter=None)

Flatten the tree by returning a depth-first search over the tree's leaves.

### get\_index\_data(index)

Get info about the specified range in the tree.

#### Returns

A dictionary with the following items:

- elem: The element in which index resides.
- *index*: Copy of the index parameter
- offset: The offset of index into 'elem'.

# get\_parent\_elem (child)

Searches the current sub-tree for and returns the parent of the child element.

#### insert (offset, text, preferred\_parent=None)

Insert the given text at the specified offset of this string-tree's string (Unicode) representation.

### insert\_between (left, right, text)

Insert the given text between the two parameter StringElems.

#### isleaf()

Whether or not this instance is a leaf node in the StringElem tree.

A node is a leaf node if it is a StringElem (not a sub-class) and contains only sub-elements of type str or unicode.

## Return type bool

### iter\_depth\_first (filter=None)

Iterate through the nodes in the tree in dept-first order.

### map (f, filter=None)

Apply f to all nodes for which filter returned True (optional).

### print\_tree (indent=0, verbose=False)

Print the tree from the current instance's point in an indented manner.

#### prune()

Remove unnecessary nodes to make the tree optimal.

#### remove\_type (ptype)

Replace nodes with type ptype with base StringElems, containing the same sub-elements. This is only applicable to elements below the element tree root node.

#### translate()

Transform the sub-tree according to some class-specific needs. This method should be either overridden in implementing sub-classes or dynamically replaced by specific applications.

**Returns** The transformed Unicode string representing the sub-tree.

# apply\_to\_strings(f)

Apply f to all actual strings in the tree.

**Parameters f** – Must take one (str or unicode) argument and return a string or unicode.

#### copy()

Returns a copy of the sub-tree. This should be overridden in sub-classes with more data.

Note: self.renderer is not copied.

### delete range (start index, end index)

Delete the text in the range given by the string-indexes start\_index and end\_index.

Partial nodes will only be removed if they are editable.

Returns A StringElem representing the removed sub-string, the parent node from which it was deleted as well as the offset at which it was deleted from. None is returned for the parent value if the root was deleted. If the parent and offset values are not None, parent. insert (offset, deleted) effectively undoes the delete.

### depth\_first (filter=None)

Returns a list of the nodes in the tree in depth-first order.

### elem\_at\_offset (offset)

Get the StringElem in the tree that contains the string rendered at the given offset.

# ${\tt elem\_offset}\;(elem)$

Find the offset of elem in the current tree.

This cannot be reliably used if self.renderer is used and even less so if the rendering function renders the string differently upon different calls. In Virtual the StringElemGUI.index() method is used as replacement for this one.

**Returns** The string index where element e starts, or -1 if e was not found.

## encode (encoding='ascii')

More unicode class emulation.

#### find(x)

Find sub-string x in this string tree and return the position at which it starts.

#### $find_elems_with(x)$

Find all elements in the current sub-tree containing x.

### flatten (filter=None)

Flatten the tree by returning a depth-first search over the tree's leaves.

### get\_index\_data(index)

Get info about the specified range in the tree.

#### Returns

A dictionary with the following items:

- elem: The element in which index resides.
- index: Copy of the index parameter
- offset: The offset of index into 'elem'.

### get\_parent\_elem(child)

Searches the current sub-tree for and returns the parent of the child element.

# insert (offset, text, preferred\_parent=None)

Insert the given text at the specified offset of this string-tree's string (Unicode) representation.

### insert\_between (left, right, text)

Insert the given text between the two parameter StringElems.

### isleaf()

Whether or not this instance is a leaf node in the StringElem tree.

A node is a leaf node if it is a StringElem (not a sub-class) and contains only sub-elements of type str or unicode.

# Return type bool

### iter\_depth\_first (filter=None)

Iterate through the nodes in the tree in dept-first order.

#### map(f, filter=None)

Apply f to all nodes for which filter returned True (optional).

## print\_tree (indent=0, verbose=False)

Print the tree from the current instance's point in an indented manner.

#### prune()

Remove unnecessary nodes to make the tree optimal.

### remove\_type (ptype)

Replace nodes with type ptype with base StringElems, containing the same sub-elements. This is only applicable to elements below the element tree root node.

#### translate()

Transform the sub-tree according to some class-specific needs. This method should be either overridden in implementing sub-classes or dynamically replaced by specific applications.

**Returns** The transformed Unicode string representing the sub-tree.

Placeable for unrecognized or unimplemented XML nodes. It's main purpose is to preserve all associated XML data.

### $apply_to_strings(f)$

Apply f to all actual strings in the tree.

**Parameters f** – Must take one (str or unicode) argument and return a string or unicode.

#### copy()

Returns a copy of the sub-tree. This should be overridden in sub-classes with more data.

Note: self.renderer is not copied.

#### delete\_range (start\_index, end\_index)

Delete the text in the range given by the string-indexes start\_index and end\_index.

Partial nodes will only be removed if they are editable.

Returns A StringElem representing the removed sub-string, the parent node from which it was deleted as well as the offset at which it was deleted from. None is returned for the parent value if the root was deleted. If the parent and offset values are not None, parent. insert (offset, deleted) effectively undoes the delete.

# depth\_first (filter=None)

Returns a list of the nodes in the tree in depth-first order.

# elem\_at\_offset (offset)

Get the StringElem in the tree that contains the string rendered at the given offset.

### elem offset(elem)

Find the offset of elem in the current tree.

This cannot be reliably used if self.renderer is used and even less so if the rendering function renders the string differently upon different calls. In Virtual the StringElemGUI.index() method is used as replacement for this one.

**Returns** The string index where element e starts, or -1 if e was not found.

#### encode (encoding='ascii')

More unicode class emulation.

### find(x)

Find sub-string x in this string tree and return the position at which it starts.

### $find_elems_with(x)$

Find all elements in the current sub-tree containing x.

#### flatten (filter=None)

Flatten the tree by returning a depth-first search over the tree's leaves.

### get\_index\_data(index)

Get info about the specified range in the tree.

#### Returns

A dictionary with the following items:

- elem: The element in which index resides.
- index: Copy of the index parameter
- offset: The offset of index into 'elem'.

#### get parent elem (child)

Searches the current sub-tree for and returns the parent of the child element.

### insert (offset, text, preferred\_parent=None)

Insert the given text at the specified offset of this string-tree's string (Unicode) representation.

#### insert\_between (left, right, text)

Insert the given text between the two parameter StringElems.

#### isleaf()

Whether or not this instance is a leaf node in the StringElem tree.

A node is a leaf node if it is a StringElem (not a sub-class) and contains only sub-elements of type str or unicode.

# Return type bool

### iter\_depth\_first (filter=None)

Iterate through the nodes in the tree in dept-first order.

### map (f, filter=None)

Apply f to all nodes for which filter returned True (optional).

### classmethod parse(pstr)

Parse an instance of this class from the start of the given string. This method should be implemented by any sub-class that wants to parseable by translate.storage.placeables.parse.

**Parameters** pstr (unicode) – The string to parse into an instance of this class.

Returns An instance of the current class, or None if the string not parseable by this class.

# print\_tree (indent=0, verbose=False)

Print the tree from the current instance's point in an indented manner.

### prune()

Remove unnecessary nodes to make the tree optimal.

# $\verb"remove_type" (ptype")$

Replace nodes with type ptype with base StringElems, containing the same sub-elements. This is only applicable to elements below the element tree root node.

#### translate()

Transform the sub-tree according to some class-specific needs. This method should be either overridden in implementing sub-classes or dynamically replaced by specific applications.

**Returns** The transformed Unicode string representing the sub-tree.

# php

Classes that hold units of PHP localisation files *phpunit* or entire files *phpfile*. These files are used in translating many PHP based applications.

Only PHP files written with these conventions are supported:

```
$lang['item'] = "vale"; # Array of values
$some_entity = "value"; # Named variables
define("ENTITY", "value");
$lang = array(
   'item1' => 'value1'
                              #Supports space before comma
   'item2' => 'value2',
);
$lang = array(
                 # Nested arrays
   'item1' => 'value1',
   'item2' => array(
      'key' => 'value'
                              #Supports space before comma
      'key2' => 'value2',
   ),
);
```

Nested arrays without key for nested array are not supported:

```
<?php
$lang = array(array('key' => 'value'));
```

The working of PHP strings and specifically the escaping conventions which differ between single quote (') and double quote (") characters are implemented as outlined in the PHP documentation for the String type.

```
translate.storage.php.phpdecode (text, quotechar=u"'")
Convert PHP escaped string to a Python string.

translate.storage.php.phpencode (text, quotechar=u"'")
Convert Python string to PHP escaping.
```

The encoding is implemented for 'single quote' and "double quote" syntax.

heredoc and nowdoc are not implemented and it is not certain whether this would ever be needed for PHP localisation needs.

```
class translate.storage.php.phpfile(inputfile=None, **kwargs)
```

This class represents a PHP file, made up of phpunits.

```
UnitClass
```

alias of phpunit

```
\verb"add_unit_to_index" (\textit{unit}")
```

Add a unit to source and location idexes

```
addsourceunit (source)
```

Add and returns a new unit with the given source string.

```
Return type TranslationUnit
```

```
addunit (unit)
```

Append the given unit to the object's list of units.

This method should always be used rather than trying to modify the list manually.

Parameters unit (TranslationUnit) - The unit that will be added.

```
detect_encoding (text, default_encodings=None)
```

Try to detect a file encoding from text, using either the chardet lib or by trying to decode the file.

```
fallback_detection (text)
```

Simple detection based on BOM in case chardet is not available.

```
findid(id)
     find unit with matching id by checking id_index
findunit (source)
     Find the unit with the given source string.
         Return type TranslationUnit or None
findunits (source)
     Find the units with the given source string.
         Return type TranslationUnit or None
getids (filename=None)
     return a list of unit ids
getprojectstyle()
     Get the project type for this store.
getsourcelanguage()
     Get the source language for this store.
gettargetlanguage()
     Get the target language for this store.
getunits()
     Return a list of all units in this store.
isempty()
     Return True if the object doesn't contain any translation units.
makeindex()
     Indexes the items in this store. At least .sourceindex should be useful.
merge_on
     The matching criterion to use when merging on.
         Returns The default matching criterion for all the subclasses.
         Return type string
parse (phpsrc)
     Read the source of a PHP file in and include them as units.
classmethod parsefile (storefile)
     Reads the given file (or opens the given filename) and parses back to an object.
classmethod parsestring(storestring)
     Convert the string representation back to an object.
remove_unit_from_index(unit)
     Remove a unit from source and locaton indexes
require_index()
     make sure source index exists
save()
     Save to the file that data was originally read from, if available.
savefile (storefile)
     Write the string representation to the given file (or filename).
serialize(out)
     Convert the units back to lines.
```

### setprojectstyle (project\_style)

Set the project type for this store.

### setsourcelanguage (sourcelanguage)

Set the source language for this store.

## settargetlanguage (targetlanguage)

Set the target language for this store.

#### translate(source)

Return the translated string for a given source string.

# Return type String or None

#### unit iter()

Iterator over all the units in this store.

# class translate.storage.php.phpunit(source=u")

A unit of a PHP file: a name, a value, and any comments associated.

### adderror (errorname, errortext)

Adds an error message to this unit.

#### **Parameters**

- **errorname** (*string*) A single word to id the error.
- **errortext** (*string*) The text describing the error.

### addlocation (location)

Add one location to the list of locations.

**Note:** Shouldn't be implemented if the format doesn't support it.

# $\verb"addlocations" (location")$

Add a location or a list of locations.

**Note:** Most classes shouldn't need to implement this, but should rather implement TranslationUnit.addlocation().

**Warning:** This method might be removed in future.

# addnote (text, origin=None, position=u'append')

Adds a note (comment).

#### **Parameters**

- text (string) Usually just a sentence or two.
- **origin** (*string*) Specifies who/where the comment comes from. Origin can be one of the following text strings: 'translator' 'developer', 'programmer', 'source code' (synonyms)

### classmethod buildfromunit (unit)

Build a native unit from a foreign unit, preserving as much information as possible.

#### getcontext()

Get the message context.

#### geterrors()

Get all error messages.

### **Return type** Dictionary

#### getid()

A unique identifier for this unit.

### Return type string

Returns an identifier for this unit that is unique in the store

Derived classes should override this in a way that guarantees a unique identifier for each unit in the store.

### getlocations()

A list of source code locations.

#### **Return type** List

Note: Shouldn't be implemented if the format doesn't support it.

#### getnotes (origin=None)

Returns all notes about this unit.

It will probably be freeform text or something reasonable that can be synthesised by the format. It should not include location comments (see <code>getlocations()</code>).

### getoutput (indent=u", name=None)

Convert the unit back into formatted lines for a php file.

### gettargetlen()

Returns the length of the target string.

## Return type Integer

**Note:** Plural forms might be combined.

#### getunits()

This unit in a list.

## hasplural()

Tells whether or not this specific unit has plural strings.

### infer\_state()

Empty method that should be overridden in sub-classes to infer the current state(\_n) of the unit from its current state.

### isblank()

Return whether this is a blank element, containing only comments.

# isfuzzy()

Indicates whether this unit is fuzzy.

#### isheader()

Indicates whether this unit is a header.

# isobsolete()

indicate whether a unit is obsolete

### isreview()

Indicates whether this unit needs review.

#### istranslatable()

Indicates whether this unit can be translated.

This should be used to distinguish real units for translation from header, obsolete, binary or other blank units.

# istranslated()

Indicates whether this unit is translated.

This should be used rather than deducing it from .target, to ensure that other classes can implement more functionality (as XLIFF does).

#### makeobsolete()

Make a unit obsolete

### markfuzzy(value=True)

Marks the unit as fuzzy or not.

### markreviewneeded (needsreview=True, explanation=None)

Marks the unit to indicate whether it needs review.

#### **Parameters**

- needsreview Defaults to True.
- **explanation** Adds an optional explanation as a note.

**merge** (otherunit, overwrite=False, comments=True, authoritative=False)

Do basic format agnostic merging.

#### multistring to rich(mulstring)

Convert a multistring to a list of "rich" string trees:

# removenotes (origin=None)

Remove all the translator's notes.

#### rich source

### See also:

```
rich_to_multistring(), multistring_to_rich()
```

# rich\_target

#### See also:

```
rich_to_multistring(), multistring_to_rich()
```

## classmethod rich\_to\_multistring(elem\_list)

Convert a "rich" string tree to a multistring:

```
>>> from translate.storage.placeables.interfaces import X
>>> rich = [StringElem(['foo', X(id='xxx', sub=[' ']), 'bar'])]
>>> TranslationUnit.rich_to_multistring(rich)
multistring(u'foo bar')
```

```
setcontext (context)
          Set the message context
     setid(value)
          Sets the unique identified for this unit.
          only implemented if format allows ids independant from other unit properties like source or context
     unit_iter()
          Iterator that only returns this unit.
translate.storage.php.wrap_production(func)
     Decorator for production functions to store lexer positions.
pocommon
translate.storage.pocommon.extract_msgid_comment(text)
     The one definitive way to extract a msgid comment out of an unescaped unicode string that might contain it.
          Return type unicode
class translate.storage.pocommon.pofile(inputfile=None, **kwargs)
     UnitClass
          alias of translate.storage.base.TranslationUnit
     add_unit_to_index(unit)
          Add a unit to source and location idexes
     addsourceunit (source)
          Add and returns a new unit with the given source string.
              Return type TranslationUnit
     addunit (unit)
          Append the given unit to the object's list of units.
          This method should always be used rather than trying to modify the list manually.
              Parameters unit (TranslationUnit) - The unit that will be added.
     detect encoding (text, default encodings=None)
          Try to detect a file encoding from text, using either the chardet lib or by trying to decode the file.
     fallback_detection (text)
          Simple detection based on BOM in case chardet is not available.
     findid(id)
          find unit with matching id by checking id_index
     findunit (source)
          Find the unit with the given source string.
              Return type TranslationUnit or None
     findunits (source)
          Find the units with the given source string.
              Return type TranslationUnit or None
     getheaderplural()
          Returns the nplural and plural values from the header.
```

#### getids (filename=None)

return a list of unit ids

### getprojectstyle()

Return the project based on information in the header.

## The project is determined in the following sequence:

- 1. Use the 'X-Project-Style' entry in the header.
- 2. Use 'Report-Msgid-Bug-To' entry
- 3. Use the 'X-Accelerator' entry
- 4. Use the Project ID
- 5. Analyse the file itself (not yet implemented)

## getsourcelanguage()

Get the source language for this store.

### gettargetlanguage()

Return the target language based on information in the header.

#### The target language is determined in the following sequence:

- 1. Use the 'Language' entry in the header.
- 2. Poedit's custom headers.
- 3. Analysing the 'Language-Team' entry.

#### getunits()

Return a list of all units in this store.

# header()

Returns the header element, or None. Only the first element is allowed to be a header. Note that this could still return an empty header element, if present.

```
init_headers (charset='UTF-8', encoding='8bit', **kwargs)
sets default values for po headers
```

# isempty()

Return True if the object doesn't contain any translation units.

```
makeheader (**kwargs)
```

Create a header for the given filename.

Check .makeheaderdict() for information on parameters.

```
makeheaderdict (charset='CHARSET', encoding='ENCODING', project_id_version=None, pot_creation_date=None, po_revision_date=None, last_translator=None, language_team=None, mime_version=None, plural_forms=None, report_msgid_bugs_to=None, **kwargs)
```

Create a header dictionary with useful defaults.

pot\_creation\_date can be None (current date) or a value (datetime or string) po\_revision\_date can be None (form), False (=pot\_creation\_date), True (=now), or a value (datetime or string)

**Returns** Dictionary with the header items

**Return type** dict of strings

### makeindex()

Indexes the items in this store. At least .sourceindex should be useful.

#### merge on

The matching criterion to use when merging on.

### mergeheaders (otherstore)

Merges another header with this header.

This header is assumed to be the template.

### parse (data)

parser to process the given source string

#### classmethod parsefile (storefile)

Reads the given file (or opens the given filename) and parses back to an object.

### parseheader()

Parses the PO header and returns the interpreted values as a dictionary.

### classmethod parsestring(storestring)

Convert the string representation back to an object.

#### remove\_unit\_from\_index(unit)

Remove a unit from source and locaton indexes

### require\_index()

make sure source index exists

#### save (

Save to the file that data was originally read from, if available.

### savefile (storefile)

Write the string representation to the given file (or filename).

### serialize (out)

Converts to a bytes representation that can be parsed back using parsestring(). out should be an open file-like objects to write to.

# setprojectstyle (project\_style)

Set the project in the header.

**Parameters** project\_style (str) - the new project

### setsourcelanguage (sourcelanguage)

Set the source language for this store.

# $\verb|settargetlanguage| (lang)$

Set the target language in the header.

This removes any custom Poedit headers if they exist.

**Parameters** lang(str) – the new target language code

## translate(source)

Return the translated string for a given source string.

Return type String or None

#### unit iter()

Iterator over all the units in this store.

### updatecontributor (name, email=None)

Add contribution comments if necessary.

#### updateheader (add=False, \*\*kwargs)

Updates the fields in the PO style header.

This will create a header if add == True.

## updateheaderplural (nplurals, plural)

Update the Plural-Form PO header.

class translate.storage.pocommon.pounit(source=None)

### adderror (errorname, errortext)

Adds an error message to this unit.

#### addlocation (location)

Add one location to the list of locations.

Note: Shouldn't be implemented if the format doesn't support it.

### addlocations (location)

Add a location or a list of locations.

**Note:** Most classes shouldn't need to implement this, but should rather implement TranslationUnit.addlocation().

**Warning:** This method might be removed in future.

addnote (text, origin=None, position='append')

Adds a note (comment).

### **Parameters**

- **text** (*string*) Usually just a sentence or two.
- **origin** (string) Specifies who/where the comment comes from. Origin can be one of the following text strings: 'translator' 'developer', 'programmer', 'source code' (synonyms)

### classmethod buildfromunit (unit)

Build a native unit from a foreign unit, preserving as much information as possible.

# getcontext()

Get the message context.

#### geterrors()

Get all error messages.

### getid()

A unique identifier for this unit.

# Return type string

**Returns** an identifier for this unit that is unique in the store

Derived classes should override this in a way that guarantees a unique identifier for each unit in the store.

#### getlocations()

A list of source code locations.

### **Return type** List

**Note:** Shouldn't be implemented if the format doesn't support it.

## getnotes (origin=None)

Returns all notes about this unit.

It will probably be freeform text or something reasonable that can be synthesised by the format. It should not include location comments (see <code>getlocations()</code>).

# gettargetlen()

Returns the length of the target string.

### Return type Integer

Note: Plural forms might be combined.

#### getunits()

This unit in a list.

### hasplural()

Tells whether or not this specific unit has plural strings.

#### infer state()

Empty method that should be overridden in sub-classes to infer the current state(\_n) of the unit from its current state.

## isblank()

Used to see if this unit has no source or target string.

**Note:** This is probably used more to find translatable units, and we might want to move in that direction rather and get rid of this.

### isfuzzy()

Indicates whether this unit is fuzzy.

### isheader()

Indicates whether this unit is a header.

# $\verb|isobsolete|()|$

indicate whether a unit is obsolete

### isreview()

Indicates whether this unit needs review.

# istranslatable()

Indicates whether this unit can be translated.

This should be used to distinguish real units for translation from header, obsolete, binary or other blank units.

#### istranslated()

Indicates whether this unit is translated.

This should be used rather than deducing it from .target, to ensure that other classes can implement more functionality (as XLIFF does).

```
makeobsolete()
          Make a unit obsolete
     markfuzzy (present=True)
          Marks the unit as fuzzy or not.
     markreviewneeded (needsreview=True, explanation=None)
          Marks the unit to indicate whether it needs review. Adds an optional explanation as a note.
     merge (otherunit, overwrite=False, comments=True, authoritative=False)
         Do basic format agnostic merging.
     multistring_to_rich(mulstring)
          Convert a multistring to a list of "rich" string trees:
          >>> target = multistring([u'foo', u'bar', u'baz'])
         >>> TranslationUnit.multistring_to_rich(target)
          [<StringElem([<StringElem([u'foo'])>])>,
           <StringElem([<StringElem([u'bar'])>])>,
           <StringElem([<StringElem([u'baz'])>])>]
     removenotes (origin=None)
          Remove all the translator's notes.
     rich_source
          See also:
          rich_to_multistring(), multistring_to_rich()
     rich_target
          See also:
          rich_to_multistring(), multistring_to_rich()
     classmethod rich_to_multistring(elem_list)
          Convert a "rich" string tree to a multistring:
          >>> from translate.storage.placeables.interfaces import X
          >>> rich = [StringElem(['foo', X(id='xxx', sub=[' ']), 'bar'])]
          >>> TranslationUnit.rich_to_multistring(rich)
         multistring(u'foo bar')
     setcontext (context)
         Set the message context
     setid(value)
          Sets the unique identified for this unit.
          only implemented if format allows ids independant from other unit properties like source or context
     unit_iter()
          Iterator that only returns this unit.
translate.storage.pocommon.quote_plus(text)
     Quote the query fragment of a URL; replacing "with "+"
translate.storage.pocommon.unquote_plus(text)
     unquote('%7e/abc+def') -> '~/abc def'
```

### poheader

class that handles all header functions for a header in a po file

```
translate.storage.poheader.parseheaderstring(input)
```

Parses an input string with the definition of a PO header and returns the interpreted values as a dictionary.

```
class translate.storage.poheader.poheader
```

This class implements functionality for manipulation of po file headers. This class is a mix-in class and useless on its own. It must be used from all classes which represent a po file

### getheaderplural()

Returns the nplural and plural values from the header.

### getprojectstyle()

Return the project based on information in the header.

## The project is determined in the following sequence:

- 1. Use the 'X-Project-Style' entry in the header.
- 2. Use 'Report-Msgid-Bug-To' entry
- 3. Use the 'X-Accelerator' entry
- 4. Use the Project ID
- 5. Analyse the file itself (not yet implemented)

#### gettargetlanguage()

Return the target language based on information in the header.

#### The target language is determined in the following sequence:

- 1. Use the 'Language' entry in the header.
- 2. Poedit's custom headers.
- 3. Analysing the 'Language-Team' entry.

#### header()

Returns the header element, or None. Only the first element is allowed to be a header. Note that this could still return an empty header element, if present.

```
init_headers (charset='UTF-8', encoding='8bit', **kwargs)
sets default values for po headers
```

```
makeheader(**kwargs)
```

Create a header for the given filename.

Check .makeheaderdict() for information on parameters.

Create a header dictionary with useful defaults.

pot\_creation\_date can be None (current date) or a value (datetime or string) po\_revision\_date can be None (form), False (=pot\_creation\_date), True (=now), or a value (datetime or string)

**Returns** Dictionary with the header items

**Return type** dict of strings

#### mergeheaders (otherstore)

Merges another header with this header.

This header is assumed to be the template.

#### parseheader()

Parses the PO header and returns the interpreted values as a dictionary.

# setprojectstyle (project\_style)

Set the project in the header.

Parameters project\_style (str) - the new project

### settargetlanguage (lang)

Set the target language in the header.

This removes any custom Poedit headers if they exist.

**Parameters** lang(str) – the new target language code

### updatecontributor (name, email=None)

Add contribution comments if necessary.

## updateheader (add=False, \*\*kwargs)

Updates the fields in the PO style header.

This will create a header if add == True.

## updateheaderplural (nplurals, plural)

Update the Plural-Form PO header.

```
translate.storage.poheader.tzstring()
```

Returns the timezone as a string in the format [+-]0000, eg +0200.

# Return type str

```
translate.storage.poheader.update(existing, add=False, **kwargs)
```

Update an existing header dictionary with the values in kwargs, adding new values only if add is true.

**Returns** Updated dictionary of header entries

Return type dict of strings

### poparser

```
translate.storage.poparser.decode header(unit, decode)
```

The header has been arbitrarily decoded with a single-byte encoding. We re-encode it to decode values with the proper encoding defined in the header (using decode\_list above).

```
translate.storage.poparser.read_obsolete_lines(parse_state)
```

Read all the lines belonging to the current unit if obsolete.

```
translate.storage.poparser.read_prevmsgid_lines(parse_state)
```

Read all the lines belonging starting with #l. These lines contain the previous msgid and msgctxt info. We strip away the leading '#l ' and read until we stop seeing #l.

### po

A class loader that will load C or Python implementations of the PO class depending on the USECPO variable.

Use the environment variable USECPO=2 (or USECPO=1) to choose the C implementation which uses Gettext's libgettextpo for high parsing speed. Otherwise the local Python based parser is used (slower but very well tested).

#### poxliff

XLIFF classes specifically suited for handling the PO representation in XLIFF.

This way the API supports plurals as if it was a PO file, for example.

```
class translate.storage.poxliff.PoXliffFile(*args, **kwargs)
    a file for the po variant of Xliff files
```

#### UnitClass

alias of PoXliffUnit

### add\_unit\_to\_index(unit)

Add a unit to source and location idexes

#### addheader()

Initialise the file header.

```
addplural (source, target, filename, createifmissing=False)
```

This method should now be unnecessary, but is left for reference

```
addsourceunit (source, filename='NoName', createifmissing=False)
```

adds the given trans-unit to the last used body node if the filename has changed it uses the slow method instead (will create the nodes required if asked). Returns success

```
addunit (unit, new=True)
```

Append the given unit to the object's list of units.

This method should always be used rather than trying to modify the list manually.

Parameters unit (TranslationUnit) - The unit that will be added.

```
createfilenode (filename, sourcelanguage='en-US', datatype='po')
```

creates a filenode with the given filename. All parameters are needed for XLIFF compliance.

```
creategroup (filename='NoName', createifmissing=False, restype=None)
adds a group tag into the specified file
```

```
detect_encoding (text, default_encodings=None)
```

Try to detect a file encoding from text, using either the chardet lib or by trying to decode the file.

### fallback detection(text)

Simple detection based on BOM in case chardet is not available.

### findid(id)

find unit with matching id by checking id\_index

## findunit (source)

Find the unit with the given source string.

Return type TranslationUnit or None

#### findunits (source)

Find the units with the given source string.

```
Return type TranslationUnit or None
```

# getbodynode (filenode, createifmissing=False)

finds the body node for the given filenode

#### getdatatype (filename=None)

Returns the datatype of the stored file. If no filename is given, the datatype of the first file is given.

# getdate (filename=None)

Returns the date attribute for the file.

If no filename is given, the date of the first file is given. If the date attribute is not specified, None is returned.

**Returns** Date attribute of file

**Return type** Date or None

#### getfilename (filenode)

returns the name of the given file

### getfilenames()

returns all filenames in this XLIFF file

### getfilenode (filename, createifmissing=False)

finds the filenode with the given name

#### getheadernode (filenode, createifmissing=False)

finds the header node for the given filenode

### getheaderplural()

Returns the nplural and plural values from the header.

#### getids (filename=None)

return a list of unit ids

# getprojectstyle()

Get the project type for this store.

## getsourcelanguage()

Get the source language for this store.

### gettargetlanguage()

Get the target language for this store.

#### getunits()

Return a list of all units in this store.

# header()

Returns the header element, or None. Only the first element is allowed to be a header. Note that this could still return an empty header element, if present.

```
init_headers (charset='UTF-8', encoding='8bit', **kwargs)
```

sets default values for po headers

#### initbody()

Initialises self.body so it never needs to be retrieved from the XML again.

## isempty()

Return True if the object doesn't contain any translation units.

### makeheader (\*\*kwargs)

Create a header for the given filename.

Check .makeheaderdict() for information on parameters.

```
makeheaderdict (charset='CHARSET', encoding='ENCODING', project_id_version=None, pot_creation_date=None, po_revision_date=None, last_translator=None, language_team=None, mime_version=None, plural_forms=None, report_msgid_bugs_to=None, **kwargs)
```

Create a header dictionary with useful defaults.

pot\_creation\_date can be None (current date) or a value (datetime or string) po\_revision\_date can be None (form), False (=pot\_creation\_date), True (=now), or a value (datetime or string)

**Returns** Dictionary with the header items

Return type dict of strings

### makeindex()

Indexes the items in this store. At least .sourceindex should be useful.

#### merge\_on

The matching criterion to use when merging on.

**Returns** The default matching criterion for all the subclasses.

Return type string

### mergeheaders (otherstore)

Merges another header with this header.

This header is assumed to be the template.

### namespaced(name)

Returns name in Clark notation.

For example namespaced ("source") in an XLIFF document might return:

```
{urn:oasis:names:tc:xliff:document:1.1}source
```

This is needed throughout lxml.

#### parse (xml)

Populates this object from the given xml string

### classmethod parsefile(storefile)

Reads the given file (or opens the given filename) and parses back to an object.

### parseheader()

Parses the PO header and returns the interpreted values as a dictionary.

# classmethod parsestring(storestring)

Parses the string to return the correct file object

### remove\_unit\_from\_index(unit)

Remove a unit from source and locaton indexes

## removedefaultfile()

We want to remove the default file-tag as soon as possible if we know if still present and empty.

#### require index()

make sure source index exists

### save()

Save to the file that data was originally read from, if available.

### savefile (storefile)

Write the string representation to the given file (or filename).

```
serialize(out)
           Converts to a string containing the file's XML
     setfilename (filenode, filename)
           set the name of the given file
     setprojectstyle (project_style)
           Set the project type for this store.
     setsourcelanguage(language)
           Set the source language for this store.
     settargetlanguage(language)
           Set the target language for this store.
     switchfile (filename, createifmissing=False)
           Adds the given trans-unit (will create the nodes required if asked).
               Returns Success
               Return type Boolean
     translate(source)
           Return the translated string for a given source string.
               Return type String or None
     unit_iter()
           Iterator over all the units in this store.
     updatecontributor (name, email=None)
           Add contribution comments if necessary.
     updateheader (add=False, **kwargs)
           Updates the fields in the PO style header.
           This will create a header if add == True.
     updateheaderplural (nplurals, plural)
           Update the Plural-Form PO header.
class translate.storage.poxliff.PoXliffUnit (source=None, empty=False, **kwargs)
     A class to specifically handle the plural units created from a po file.
     addalttrans (txt, origin=None, lang=None, sourcetxt=None, matchquality=None)
           Adds an alt-trans tag and alt-trans components to the unit.
               Parameters txt (String) – Alternative translation of the source text.
     adderror (errorname, errortext)
           Adds an error message to this unit.
     addlocation (location)
           Add one location to the list of locations.
           Note: Shouldn't be implemented if the format doesn't support it.
```

4.1. API 563

addlocations (location)

Add a location or a list of locations.

**Note:** Most classes shouldn't need to implement this, but should rather implement TranslationUnit.addlocation().

**Warning:** This method might be removed in future.

### addnote (text, origin=None, position='append')

Add a note specifically in a "note" tag

### classmethod buildfromunit (unit)

Build a native unit from a foreign unit, preserving as much information as possible.

#### correctorigin (node, origin)

Check against node tag's origin (e.g note or alt-trans)

### createcontextgroup (name, contexts=None, purpose=None)

Add the context group to the trans-unit with contexts a list with (type, text) tuples describing each context.

#### createlanguageNode (lang, text, purpose)

Returns an xml Element setup with given parameters.

#### delalttrans (alternative)

Removes the supplied alternative from the list of alt-trans tags

# getNodeText (languageNode, xml\_space='preserve')

Retrieves the term from the given languageNode.

#### get\_rich\_target (lang=None)

retrieves the "target" text (second entry), or the entry in the specified language, if it exists

### getalttrans (origin=None)

Returns <alt-trans> for the given origin as a list of units. No origin means all alternatives.

#### getautomaticcomments()

Returns the automatic comments (x-po-autocomment), which corresponds to the #. style po comments.

### getcontext()

Get the message context.

### getcontextgroups (name)

Returns the contexts in the context groups with the specified name

### geterrors()

Get all error messages.

#### getid(

A unique identifier for this unit.

### Return type string

Returns an identifier for this unit that is unique in the store

Derived classes should override this in a way that guarantees a unique identifier for each unit in the store.

### getlanguageNode (lang=None, index=None)

Retrieves a languageNode either by language or by index.

### getlanguageNodes()

We override this to get source and target nodes.

#### getlocations()

Returns all the references (source locations)

### getnotes (origin=None)

Returns all notes about this unit.

It will probably be freeform text or something reasonable that can be synthesised by the format. It should not include location comments (see <code>qetlocations()</code>).

#### getrestype()

returns the restype attribute in the trans-unit tag

### gettarget (lang=None)

retrieves the "target" text (second entry), or the entry in the specified language, if it exists

#### gettargetlen()

Returns the length of the target string.

Return type Integer

Note: Plural forms might be combined.

### gettranslatorcomments()

Returns the translator comments (x-po-trancomment), which corresponds to the # style po comments.

#### getunits()

This unit in a list.

#### hasplural()

Tells whether or not this specific unit has plural strings.

### infer\_state()

Empty method that should be overridden in sub-classes to infer the current state(\_n) of the unit from its current state.

## isapproved()

States whether this unit is approved.

#### isblank()

Used to see if this unit has no source or target string.

**Note:** This is probably used more to find translatable units, and we might want to move in that direction rather and get rid of this.

#### isfuzzy()

Indicates whether this unit is fuzzy.

### isheader()

Indicates whether this unit is a header.

### isobsolete()

indicate whether a unit is obsolete

#### isreview()

States whether this unit needs to be reviewed

## istranslatable()

Indicates whether this unit can be translated.

This should be used to distinguish real units for translation from header, obsolete, binary or other blank units.

### istranslated()

Indicates whether this unit is translated.

This should be used rather than deducing it from .target, to ensure that other classes can implement more functionality (as XLIFF does).

#### makeobsolete()

Make a unit obsolete

# markapproved (value=True)

Mark this unit as approved.

#### markfuzzy(value=True)

Marks the unit as fuzzy or not.

### markreviewneeded (needsreview=True, explanation=None)

Marks the unit to indicate whether it needs review.

Adds an optional explanation as a note.

### merge (otherunit, overwrite=False, comments=True, authoritative=False)

Do basic format agnostic merging.

### multistring\_to\_rich (mulstring)

Convert a multistring to a list of "rich" string trees:

### namespaced(name)

Returns name in Clark notation.

For example namespaced ("source") in an XLIFF document might return:

```
{urn:oasis:names:tc:xliff:document:1.1}source
```

This is needed throughout lxml.

## removenotes (origin=None)

Remove all the translator notes.

## rich\_source

#### See also:

```
rich to multistring(), multistring to rich()
```

#### rich\_target

### See also:

```
rich_to_multistring(), multistring_to_rich()
```

# $\verb|classmethod| \verb|rich_to_multistring| (elem_list)$

Convert a "rich" string tree to a multistring:

```
>>> from translate.storage.placeables.interfaces import X
          >>> rich = [StringElem(['foo', X(id='xxx', sub=[' ']), 'bar'])]
          >>> TranslationUnit.rich_to_multistring(rich)
          multistring(u'foo bar')
     setcontext (context)
          Set the message context
     setid(id)
          Sets the unique identified for this unit.
          only implemented if format allows ids independant from other unit properties like source or context
     settarget (target, lang='xx', append=False)
          Sets the target string to the given value.
     unit iter()
          Iterator that only returns this unit.
project
class translate.storage.project.Project(projstore=None)
     Manages a project store as well as the processes involved in a project workflow.
     add_source (srcfile, src_fname=None)
          Proxy for self.store.append_sourcefile().
     add_source_convert (srcfile, src_fname=None, convert_options=None, extension=None)
          Convenience method that calls add_source() and convert_forward() and returns the results
          from both.
     close()
          Proxy for self.store.close().
     convert_forward (input_fname, template=None, output_fname=None, **options)
          Convert the given input file to the next type in the process:
          Source document (eg. ODT) -> Translation file (eg. XLIFF) -> Translated document (eg. ODT).
              Parameters
                  • input fname (basestring) - The project name of the file to convert
                  • convert_options (Dictionary (optional)) - Passed as-is to translate.
                    convert.factory.convert().
              Returns 2-tuple the converted file object and its project name.
     export_file (fname, destfname)
          Export the file with the specified filename to the given destination.
                                                                               This method will raise
          FileNotInProjectError via the call to get_file() if fname is not found in the project.
     get file(fname)
          Proxy for self.store.get_file().
     get_proj_filename (realfname)
          Proxy for self.store.get proj filename().
     get_real_filename (projfname)
          Try and find a real file name for the given project file name.
```

```
remove_file (projfname, ftype=None)
          Proxy for self.store.remove file().
     save (filename=None)
          Proxy for self.store.save().
     update_file (proj_fname, infile)
          Proxy for self.store.update file().
projstore
exception translate.storage.projstore.FileExistsInProjectError
exception translate.storage.projstore.FileNotInProjectError
class translate.storage.projstore.ProjectStore
     Basic project file container.
     append_file (afile, fname, ftype='trans', delete_orig=False)
          Append the given file to the project with the given filename, marked to be of type ftype ('src', 'trans',
          'tgt').
              Parameters delete_orig(bool) - Whether or not the original (given) file should be deleted
                  after being appended. This is set to True by convert_forward(). Not used in this
                  class.
     get file (fname, mode='rb')
          Retrieve the file with the given name from the project store.
          The file is looked up in the self._files dictionary. The values in this dictionary may be None, to
          indicate that the file is not cacheable and needs to be retrieved in a special way. This special way must be
          defined in this method of sub-classes. The value may also be a string, which indicates that it is a real file
          accessible via open.
              Parameters mode (str) – The mode in which to re-open the file (if it is closed).
     get_filename_type (fname)
          Get the type of file ('src', 'trans', 'tgt') with the given name.
     get proj filename(realfname)
          Try and find a project file name for the given real file name.
     load(*args, **kwargs)
          Load the project in some way. Undefined for this (base) class.
     remove file (fname, ftype=None)
          Remove the file with the given project name from the project. If the file type ('src', 'trans' or 'tgt') is not
          given, it is guessed.
     save (filename=None, *args, **kwargs)
          Save the project in some way. Undefined for this (base) class.
     sourcefiles
          Read-only access to self._sourcefiles.
     targetfiles
          Read-only access to self._targetfiles.
     transfiles
```

Read-only access to self. transfiles.

```
update_file (pfname, infile)
```

Remove the project file with name pfname and add the contents from infile to the project under the same file name.

**Returns** the results from *ProjectStore.append\_file()*.

### properties

Classes that hold units of .properties, and similar, files that are used in translating Java, Mozilla, MacOS and other software.

The propfile class is a monolingual class with propunit providing unit level access.

The .properties store has become a general key value pair class with <code>Dialect</code> providing the ability to change the behaviour of the parsing and handling of the various dialects.

Currently we support:

- · Java .properties
- · Mozilla .properties
- · Adobe Flex files
- · MacOS X .strings files
- · Skype .lang files

The following provides references and descriptions of the various dialects supported:

Java Java properties are supported completely except for the ability to drop pairs that are not translated.

The following .properties file description gives a good references to the .properties specification.

Properties file may also hold Java MessageFormat messages. No special handling is provided in this storage class for MessageFormat, but this may be implemented in future.

All delimiter types, comments, line continuations and spaces handling in delimeters are supported.

**Mozilla** Mozilla files use '=' as a delimiter, are UTF-8 encoded and thus don't need u escaping. Any U values will be converted to correct Unicode characters.

**Strings** Mac OS X strings files are implemented using these two articles as references.

**Flex** Adobe Flex files seem to be normal .properties files but in UTF-8 just like Mozilla files. This page provides the information used to implement the dialect.

**Skype** Skype .lang files seem to be UTF-16 encoded .properties files.

A simple summary of what is permissible follows.

Comments supported:

```
# a comment
// a comment (only at the beginning of a line)

# The following are # escaped to render in docs
# ! is standard but not widely supported
#! a comment
# /* is non-standard but used on some implementations
#/* a comment (not across multiple lines) */
```

Name and Value pairs:

```
# Delimiters
key = value
key : value
# Whitespace delimiter
# key[sp]value
# Space in key and around value
# Note that the b and c are escaped for reST rendering
c = a string with a continuation line \
   continuation line
# Special cases
# key with no value
//key (escaped; doesn't render in docs)
# value no key (extractable in prop2po but not mergeable in po2prop)
=value
# .strings specific
"key" = "value";
```

### class translate.storage.properties.Dialect

Settings for the various behaviours in key=value files.

classmethod encode(string, encoding=None)

Encode the string

### classmethod find\_delimiter(line)

Find the type and position of the delimiter in a property line.

Property files can be delimited by "=", ":" or whitespace (space for now). We find the position of each delimiter, then find the one that appears first.

### **Parameters**

- line (str) A properties line
- delimiters (list) valid delimiters

Returns delimiter character and offset within line

**Return type** Tuple (delimiter char, Offset Integer)

```
classmethod key_strip(key)
```

Strip unneeded characters from the key

### classmethod value\_strip(value)

Strip unneeded characters from the value

class translate.storage.properties.DialectFlex

```
classmethod encode (string, encoding=None)
```

Encode the string

## classmethod find\_delimiter(line)

Find the type and position of the delimiter in a property line.

Property files can be delimited by "=", ":" or whitespace (space for now). We find the position of each delimiter, then find the one that appears first.

### **Parameters**

- line (str) A properties line
- **delimiters** (list) valid delimiters

Returns delimiter character and offset within line

**Return type** Tuple (delimiter char, Offset Integer)

### classmethod key\_strip(key)

Strip unneeded characters from the key

# classmethod value\_strip(value)

Strip unneeded characters from the value

class translate.storage.properties.DialectGaia

## classmethod encode (string, encoding=None)

Encode the string

### classmethod find delimiter(line)

Find the type and position of the delimiter in a property line.

Property files can be delimited by "=", ":" or whitespace (space for now). We find the position of each delimiter, then find the one that appears first.

#### **Parameters**

- line (str) A properties line
- **delimiters** (list) valid delimiters

Returns delimiter character and offset within line

Return type Tuple (delimiter char, Offset Integer)

# classmethod key\_strip(key)

Strip unneeded characters from the key

# classmethod value\_strip(value)

Strip unneeded characters from the value

class translate.storage.properties.DialectJava

## classmethod encode (string, encoding=None)

Encode the string

### classmethod find delimiter(line)

Find the type and position of the delimiter in a property line.

Property files can be delimited by "=", ":" or whitespace (space for now). We find the position of each delimiter, then find the one that appears first.

#### **Parameters**

- line (str) A properties line
- delimiters (list) valid delimiters

Returns delimiter character and offset within line

Return type Tuple (delimiter char, Offset Integer)

#### classmethod key\_strip(key)

Strip unneeded characters from the key

### classmethod value\_strip(value)

Strip unneeded characters from the value

class translate.storage.properties.DialectJavaUtf8

#### classmethod encode (string, encoding=None)

Encode the string

### classmethod find\_delimiter(line)

Find the type and position of the delimiter in a property line.

Property files can be delimited by "=", ":" or whitespace (space for now). We find the position of each delimiter, then find the one that appears first.

#### **Parameters**

- line (str) A properties line
- **delimiters** (list) valid delimiters

**Returns** delimiter character and offset within *line* 

**Return type** Tuple (delimiter char, Offset Integer)

### classmethod key\_strip(key)

Strip unneeded characters from the key

### classmethod value\_strip(value)

Strip unneeded characters from the value

class translate.storage.properties.DialectJoomla

# classmethod encode (string, encoding=None)

Encode the string

### classmethod find\_delimiter(line)

Find the type and position of the delimiter in a property line.

Property files can be delimited by "=", ":" or whitespace (space for now). We find the position of each delimiter, then find the one that appears first.

#### **Parameters**

- line (str) A properties line
- **delimiters** (list) valid delimiters

**Returns** delimiter character and offset within *line* 

**Return type** Tuple (delimiter char, Offset Integer)

### classmethod key\_strip(key)

Strip unneeded characters from the key

#### classmethod value\_strip(value)

Strip unneeded characters from the value

class translate.storage.properties.DialectMozilla

#### classmethod encode (string, encoding=None)

Encode the string

#### classmethod find delimiter(line)

Find the type and position of the delimiter in a property line.

Property files can be delimited by "=", ":" or whitespace (space for now). We find the position of each delimiter, then find the one that appears first.

#### **Parameters**

- line (str) A properties line
- delimiters (list) valid delimiters

**Returns** delimiter character and offset within *line* 

**Return type** Tuple (delimiter char, Offset Integer)

### classmethod key\_strip(key)

Strip unneeded characters from the key

## classmethod value\_strip(value)

Strip unneeded characters from the value

```
class translate.storage.properties.DialectSkype
```

## classmethod encode (string, encoding=None)

Encode the string

#### classmethod find delimiter(line)

Find the type and position of the delimiter in a property line.

Property files can be delimited by "=", ":" or whitespace (space for now). We find the position of each delimiter, then find the one that appears first.

#### **Parameters**

- line (str) A properties line
- delimiters (list) valid delimiters

Returns delimiter character and offset within line

Return type Tuple (delimiter char, Offset Integer)

## ${\tt classmethod\ key\_strip}\,(\textit{key})$

Strip unneeded characters from the key

## classmethod value\_strip(value)

Strip unneeded characters from the value

```
class translate.storage.properties.DialectStrings
```

## classmethod encode (string, encoding=None)

Encode the string

#### classmethod find\_delimiter(line)

Find the type and position of the delimiter in a property line.

Property files can be delimited by "=", ":" or whitespace (space for now). We find the position of each delimiter, then find the one that appears first.

#### **Parameters**

```
• line (str) - A properties line
```

• **delimiters** (*list*) – valid delimiters

Returns delimiter character and offset within line

Return type Tuple (delimiter char, Offset Integer)

#### classmethod key strip(key)

Strip unneeded characters from the key

## classmethod value\_strip(value)

Strip unneeded characters from the value

class translate.storage.properties.DialectStringsUtf8

#### classmethod encode (string, encoding=None)

Encode the string

## classmethod find\_delimiter(line)

Find the type and position of the delimiter in a property line.

Property files can be delimited by "=", ":" or whitespace (space for now). We find the position of each delimiter, then find the one that appears first.

#### **Parameters**

- line (str) A properties line
- **delimiters** (list) valid delimiters

**Returns** delimiter character and offset within *line* 

**Return type** Tuple (delimiter char, Offset Integer)

## classmethod key\_strip(key)

Strip unneeded characters from the key

## classmethod value\_strip(value)

Strip unneeded characters from the value

```
translate.storage.properties.accesskeysuffixes = ('.accesskey', '.accessKey', '.akey')

Accesskey Suffixes: entries with this suffix may be combined with labels ending in labelsuffixes into
```

accelerator notation

```
\verb|translate.storage.properties.is_comment_end|(line)
```

Determine whether a *line* ends a new multi-line comment.

Parameters line (unicode) – A properties line

Returns True if line ends a new multi-line comment

Return type bool

```
translate.storage.properties.is_comment_one_line(line)
```

Determine whether a *line* is a one-line comment.

Parameters line (unicode) - A properties line

**Returns** True if line is a one-line comment

Return type bool

```
translate.storage.properties.is_comment_start(line)
```

Determine whether a *line* starts a new multi-line comment.

```
Parameters line (unicode) – A properties line
          Returns True if line starts a new multi-line comment
          Return type bool
translate.storage.properties.is_line_continuation(line)
     Determine whether line has a line continuation marker.
      properties files can be terminated with a backslash () indicating that the 'value' continues on the next line.
     Continuation is only valid if there are an odd number of backslashses (an even number would result in a set of
     N/2 slashes not an escape)
          Parameters line (str) – A properties line
          Returns Does line end with a line continuation
          Return type Boolean
class translate.storage.properties.javafile(*args, **kwargs)
     UnitClass
          alias of propunit
     add_unit_to_index(unit)
          Add a unit to source and location idexes
     addsourceunit (source)
          Add and returns a new unit with the given source string.
              Return type TranslationUnit
     addunit (unit)
          Append the given unit to the object's list of units.
          This method should always be used rather than trying to modify the list manually.
              Parameters unit (TranslationUnit) – The unit that will be added.
     detect_encoding (text, default_encodings=None)
          Try to detect a file encoding from text, using either the chardet lib or by trying to decode the file.
     fallback detection (text)
          Simple detection based on BOM in case chardet is not available.
     findid(id)
          find unit with matching id by checking id_index
     findunit (source)
          Find the unit with the given source string.
              Return type TranslationUnit or None
     findunits (source)
          Find the units with the given source string.
               Return type TranslationUnit or None
```

4.1. API 575

getids (filename=None)
 return a list of unit ids

getprojectstyle()

Get the project type for this store.

```
getsourcelanguage()
           Get the source language for this store.
     gettargetlanguage()
           Get the target language for this store.
     getunits()
          Return a list of all units in this store.
     isempty()
          Return True if the object doesn't contain any translation units.
     makeindex()
           Indexes the items in this store. At least .sourceindex should be useful.
     merge_on
           The matching criterion to use when merging on.
               Returns The default matching criterion for all the subclasses.
               Return type string
     parse (propsrc)
           Read the source of a properties file in and include them as units.
     classmethod parsefile(storefile)
           Reads the given file (or opens the given filename) and parses back to an object.
     classmethod parsestring(storestring)
           Convert the string representation back to an object.
     remove_unit_from_index(unit)
           Remove a unit from source and locaton indexes
     require_index()
           make sure source index exists
     save()
           Save to the file that data was originally read from, if available.
     savefile (storefile)
           Write the string representation to the given file (or filename).
     serialize (out)
           Write the units back to file.
     setprojectstyle (project_style)
           Set the project type for this store.
     setsourcelanguage (sourcelanguage)
           Set the source language for this store.
     settargetlanguage (targetlanguage)
           Set the target language for this store.
     translate(source)
           Return the translated string for a given source string.
               Return type String or None
     unit iter()
           Iterator over all the units in this store.
class translate.storage.properties.javautf8file(*args, **kwargs)
```

```
UnitClass
     alias of propunit
add_unit_to_index(unit)
     Add a unit to source and location idexes
addsourceunit (source)
     Add and returns a new unit with the given source string.
         Return type TranslationUnit
addunit (unit)
     Append the given unit to the object's list of units.
     This method should always be used rather than trying to modify the list manually.
         Parameters unit (TranslationUnit) – The unit that will be added.
detect_encoding(text, default_encodings=None)
     Try to detect a file encoding from text, using either the chardet lib or by trying to decode the file.
fallback detection(text)
     Simple detection based on BOM in case chardet is not available.
findid(id)
     find unit with matching id by checking id_index
findunit (source)
     Find the unit with the given source string.
         Return type TranslationUnit or None
findunits (source)
     Find the units with the given source string.
         Return type TranslationUnit or None
getids (filename=None)
     return a list of unit ids
getprojectstyle()
     Get the project type for this store.
getsourcelanguage()
     Get the source language for this store.
gettargetlanguage()
     Get the target language for this store.
getunits()
     Return a list of all units in this store.
isempty()
     Return True if the object doesn't contain any translation units.
makeindex()
     Indexes the items in this store. At least .sourceindex should be useful.
```

The matching criterion to use when merging on.

Return type string

**Returns** The default matching criterion for all the subclasses.

```
parse (propsrc)
           Read the source of a properties file in and include them as units.
     classmethod parsefile(storefile)
           Reads the given file (or opens the given filename) and parses back to an object.
     classmethod parsestring(storestring)
           Convert the string representation back to an object.
     remove unit from index(unit)
           Remove a unit from source and locaton indexes
     require_index()
          make sure source index exists
     save()
           Save to the file that data was originally read from, if available.
     savefile (storefile)
           Write the string representation to the given file (or filename).
     serialize(out)
           Write the units back to file.
     setprojectstyle (project_style)
           Set the project type for this store.
     setsourcelanguage (sourcelanguage)
           Set the source language for this store.
     settargetlanguage (targetlanguage)
           Set the target language for this store.
     translate(source)
           Return the translated string for a given source string.
               Return type String or None
     unit_iter()
          Iterator over all the units in this store.
class translate.storage.properties.joomlafile(*args, **kwargs)
     UnitClass
           alias of propunit
     add_unit_to_index(unit)
           Add a unit to source and location idexes
     addsourceunit (source)
           Add and returns a new unit with the given source string.
               Return type TranslationUnit
     addunit (unit)
           Append the given unit to the object's list of units.
           This method should always be used rather than trying to modify the list manually.
               Parameters unit (TranslationUnit) - The unit that will be added.
     detect encoding(text, default encodings=None)
           Try to detect a file encoding from text, using either the chardet lib or by trying to decode the file.
```

```
fallback detection(text)
     Simple detection based on BOM in case chardet is not available.
findid(id)
     find unit with matching id by checking id_index
findunit (source)
     Find the unit with the given source string.
         Return type TranslationUnit or None
findunits (source)
     Find the units with the given source string.
         Return type TranslationUnit or None
getids (filename=None)
     return a list of unit ids
getprojectstyle()
     Get the project type for this store.
getsourcelanguage()
     Get the source language for this store.
gettargetlanguage()
     Get the target language for this store.
getunits()
     Return a list of all units in this store.
isempty()
     Return True if the object doesn't contain any translation units.
makeindex()
     Indexes the items in this store. At least .sourceindex should be useful.
merge_on
     The matching criterion to use when merging on.
         Returns The default matching criterion for all the subclasses.
         Return type string
parse (propsrc)
     Read the source of a properties file in and include them as units.
classmethod parsefile(storefile)
     Reads the given file (or opens the given filename) and parses back to an object.
classmethod parsestring(storestring)
     Convert the string representation back to an object.
remove_unit_from_index(unit)
     Remove a unit from source and locaton indexes
require_index()
     make sure source index exists
save()
     Save to the file that data was originally read from, if available.
savefile (storefile)
```

4.1. API 579

Write the string representation to the given file (or filename).

```
serialize(out)
          Write the units back to file.
     setprojectstyle (project_style)
          Set the project type for this store.
     setsourcelanguage (sourcelanguage)
          Set the source language for this store.
     settargetlanguage (targetlanguage)
          Set the target language for this store.
     translate(source)
          Return the translated string for a given source string.
               Return type String or None
     unit_iter()
          Iterator over all the units in this store.
translate.storage.properties.labelsuffixes = ('.label', '.title')
     Label suffixes: entries with this suffix are able to be comibed with accesskeys found in entries ending with
     accesskeysuffixes
class translate.storage.properties.propfile(inputfile=None, personality='java', encod-
                                                            ing=None)
     this class represents a .properties file, made up of propunits
     UnitClass
          alias of propunit
     add unit to index (unit)
          Add a unit to source and location idexes
     addsourceunit (source)
          Add and returns a new unit with the given source string.
               Return type TranslationUnit
     addunit (unit)
          Append the given unit to the object's list of units.
          This method should always be used rather than trying to modify the list manually.
              Parameters unit (TranslationUnit) - The unit that will be added.
     detect encoding(text, default encodings=None)
          Try to detect a file encoding from text, using either the chardet lib or by trying to decode the file.
     fallback_detection (text)
          Simple detection based on BOM in case chardet is not available.
     findid(id)
          find unit with matching id by checking id_index
     findunit (source)
          Find the unit with the given source string.
               Return type TranslationUnit or None
     findunits (source)
          Find the units with the given source string.
               Return type TranslationUnit or None
```

```
getids (filename=None)
     return a list of unit ids
getprojectstyle()
     Get the project type for this store.
getsourcelanguage()
     Get the source language for this store.
gettargetlanguage()
     Get the target language for this store.
getunits()
     Return a list of all units in this store.
isempty()
     Return True if the object doesn't contain any translation units.
makeindex()
     Indexes the items in this store. At least .sourceindex should be useful.
merge on
     The matching criterion to use when merging on.
         Returns The default matching criterion for all the subclasses.
         Return type string
parse (propsrc)
     Read the source of a properties file in and include them as units.
classmethod parsefile(storefile)
     Reads the given file (or opens the given filename) and parses back to an object.
classmethod parsestring(storestring)
     Convert the string representation back to an object.
remove_unit_from_index(unit)
     Remove a unit from source and locaton indexes
require_index()
     make sure source index exists
save()
     Save to the file that data was originally read from, if available.
savefile (storefile)
     Write the string representation to the given file (or filename).
serialize(out)
     Write the units back to file.
setprojectstyle (project_style)
     Set the project type for this store.
setsourcelanguage (sourcelanguage)
     Set the source language for this store.
settargetlanguage (targetlanguage)
     Set the target language for this store.
translate (source)
     Return the translated string for a given source string.
```

4.1. API 581

**Return type** String or None

#### unit iter()

Iterator over all the units in this store.

## class translate.storage.properties.propunit (source=", personality='java')

An element of a properties file i.e. a name and value, and any comments associated.

## adderror (errorname, errortext)

Adds an error message to this unit.

#### **Parameters**

- **errorname** (*string*) A single word to id the error.
- **errortext** (*string*) The text describing the error.

#### addlocation (location)

Add one location to the list of locations.

**Note:** Shouldn't be implemented if the format doesn't support it.

### addlocations (location)

Add a location or a list of locations.

**Note:** Most classes shouldn't need to implement this, but should rather implement TranslationUnit.addlocation().

**Warning:** This method might be removed in future.

## addnote (text, origin=None, position='append')

Adds a note (comment).

### **Parameters**

- text (string) Usually just a sentence or two.
- **origin** (*string*) Specifies who/where the comment comes from. Origin can be one of the following text strings: 'translator' 'developer', 'programmer', 'source code' (synonyms)

## classmethod buildfromunit (unit)

Build a native unit from a foreign unit, preserving as much information as possible.

## getcontext()

Get the message context.

## geterrors()

Get all error messages.

## Return type Dictionary

## getid()

A unique identifier for this unit.

## Return type string

Returns an identifier for this unit that is unique in the store

Derived classes should override this in a way that guarantees a unique identifier for each unit in the store.

## getlocations()

A list of source code locations.

## Return type List

Note: Shouldn't be implemented if the format doesn't support it.

#### getnotes (origin=None)

Returns all notes about this unit.

It will probably be freeform text or something reasonable that can be synthesised by the format. It should not include location comments (see getlocations()).

#### getoutput()

Convert the element back into formatted lines for a .properties file

## gettargetlen()

Returns the length of the target string.

## Return type Integer

Note: Plural forms might be combined.

#### getunits()

This unit in a list.

#### hasplural()

Tells whether or not this specific unit has plural strings.

## infer\_state()

Empty method that should be overridden in sub-classes to infer the current state(\_n) of the unit from its current state.

#### isblank()

returns whether this is a blank element, containing only comments.

#### isfuzzy()

Indicates whether this unit is fuzzy.

#### isheader()

Indicates whether this unit is a header.

#### isobsolete()

indicate whether a unit is obsolete

#### isreview()

Indicates whether this unit needs review.

## istranslatable()

Indicates whether this unit can be translated.

This should be used to distinguish real units for translation from header, obsolete, binary or other blank units.

## istranslated()

Indicates whether this unit is translated.

This should be used rather than deducing it from .target, to ensure that other classes can implement more functionality (as XLIFF does).

```
makeobsolete()
```

Make a unit obsolete

## markfuzzy(value=True)

Marks the unit as fuzzy or not.

## markreviewneeded (needsreview=True, explanation=None)

Marks the unit to indicate whether it needs review.

#### **Parameters**

- needsreview Defaults to True.
- **explanation** Adds an optional explanation as a note.

merge (other unit, overwrite = False, comments = True, authoritative = False)

Do basic format agnostic merging.

## multistring\_to\_rich (mulstring)

Convert a multistring to a list of "rich" string trees:

#### removenotes (origin=None)

Remove all the translator's notes.

#### rich source

### See also:

```
rich to multistring(), multistring to rich()
```

#### rich\_target

### See also:

```
rich_to_multistring(), multistring_to_rich()
```

#### classmethod rich\_to\_multistring(elem\_list)

Convert a "rich" string tree to a multistring:

```
>>> from translate.storage.placeables.interfaces import X
>>> rich = [StringElem(['foo', X(id='xxx', sub=[' ']), 'bar'])]
>>> TranslationUnit.rich_to_multistring(rich)
multistring(u'foo bar')
```

### setcontext (context)

Set the message context

## setid(value)

Sets the unique identified for this unit.

only implemented if format allows ids independant from other unit properties like source or context

## unit\_iter()

Iterator that only returns this unit.

```
translate.storage.properties.register_dialect(dialect)
     Decorator that registers the dialect.
class translate.storage.properties.stringsfile(*args, **kwargs)
     UnitClass
          alias of propunit
     add_unit_to_index(unit)
          Add a unit to source and location idexes
     addsourceunit (source)
          Add and returns a new unit with the given source string.
               Return type TranslationUnit
     addunit (unit)
          Append the given unit to the object's list of units.
          This method should always be used rather than trying to modify the list manually.
              Parameters unit (TranslationUnit) - The unit that will be added.
     detect_encoding(text, default_encodings=None)
          Try to detect a file encoding from text, using either the chardet lib or by trying to decode the file.
     fallback_detection (text)
          Simple detection based on BOM in case chardet is not available.
     findid(id)
          find unit with matching id by checking id_index
     findunit (source)
          Find the unit with the given source string.
               Return type TranslationUnit or None
     findunits (source)
          Find the units with the given source string.
               Return type TranslationUnit or None
     getids (filename=None)
          return a list of unit ids
     getprojectstyle()
          Get the project type for this store.
     getsourcelanguage()
          Get the source language for this store.
     gettargetlanguage()
          Get the target language for this store.
     getunits()
          Return a list of all units in this store.
          Return True if the object doesn't contain any translation units.
     makeindex()
```

4.1. API 585

Indexes the items in this store. At least .sourceindex should be useful.

merge on

```
The matching criterion to use when merging on.
               Returns The default matching criterion for all the subclasses.
               Return type string
     parse (propsrc)
          Read the source of a properties file in and include them as units.
     classmethod parsefile(storefile)
          Reads the given file (or opens the given filename) and parses back to an object.
     classmethod parsestring(storestring)
          Convert the string representation back to an object.
     remove_unit_from_index(unit)
          Remove a unit from source and locaton indexes
     require_index()
          make sure source index exists
     save()
          Save to the file that data was originally read from, if available.
     savefile (storefile)
          Write the string representation to the given file (or filename).
     serialize(out)
          Write the units back to file.
     setprojectstyle (project_style)
          Set the project type for this store.
     setsourcelanguage (sourcelanguage)
          Set the source language for this store.
     settargetlanguage (targetlanguage)
          Set the target language for this store.
     translate(source)
          Return the translated string for a given source string.
               Return type String or None
     unit iter()
          Iterator over all the units in this store.
class translate.storage.properties.stringsutf8file(*args, **kwargs)
     UnitClass
          alias of propunit
     add_unit_to_index(unit)
          Add a unit to source and location idexes
     addsourceunit (source)
          Add and returns a new unit with the given source string.
               Return type TranslationUnit
     addunit (unit)
          Append the given unit to the object's list of units.
```

This method should always be used rather than trying to modify the list manually.

**Parameters unit** (TranslationUnit) – The unit that will be added.

## detect\_encoding(text, default\_encodings=None)

Try to detect a file encoding from *text*, using either the chardet lib or by trying to decode the file.

#### fallback\_detection (text)

Simple detection based on BOM in case chardet is not available.

#### findid(id)

find unit with matching id by checking id\_index

## findunit (source)

Find the unit with the given source string.

Return type TranslationUnit or None

#### findunits (source)

Find the units with the given source string.

Return type TranslationUnit or None

#### getids (filename=None)

return a list of unit ids

## getprojectstyle()

Get the project type for this store.

### getsourcelanguage()

Get the source language for this store.

## gettargetlanguage()

Get the target language for this store.

## getunits()

Return a list of all units in this store.

## isempty()

Return True if the object doesn't contain any translation units.

#### makeindex()

Indexes the items in this store. At least .sourceindex should be useful.

## merge\_on

The matching criterion to use when merging on.

**Returns** The default matching criterion for all the subclasses.

Return type string

#### parse (propsrc)

Read the source of a properties file in and include them as units.

## classmethod parsefile (storefile)

Reads the given file (or opens the given filename) and parses back to an object.

#### classmethod parsestring(storestring)

Convert the string representation back to an object.

### remove\_unit\_from\_index(unit)

Remove a unit from source and locaton indexes

## require\_index()

make sure source index exists

```
save()
          Save to the file that data was originally read from, if available.
     savefile (storefile)
          Write the string representation to the given file (or filename).
     serialize(out)
          Write the units back to file.
     setprojectstyle (project_style)
          Set the project type for this store.
     setsourcelanguage (sourcelanguage)
          Set the source language for this store.
     settargetlanguage (targetlanguage)
          Set the target language for this store.
     translate(source)
          Return the translated string for a given source string.
               Return type String or None
     unit iter()
          Iterator over all the units in this store.
pypo
Classes that hold units of Gettext .po files (pounit) or entire files (pofile).
translate.storage.pypo.escapeforpo(line)
     Escapes a line for po format. assumes no occurs in the line.
               param line unescaped text
translate.storage.pypo.extractpoline(*args, **kwargs)
     Remove quote and unescape line from po file.
          Parameters line – a quoted line from a po file (msgid or msgstr)
     Deprecated since version 1.10: Replaced by unescape(). extractpoline() is kept to allow tests of
     correctness, and in case of external users.
translate.storage.pypo.lsep = '\n#:
     Separator for #: entries
class translate.storage.pypo.pofile (inputfile=None, width=None, **kwargs)
     A .po file containing various units
     UnitClass
          alias of pounit
     add_unit_to_index(unit)
          Add a unit to source and location idexes
     addsourceunit (source)
          Add and returns a new unit with the given source string.
               Return type TranslationUnit
     addunit (unit)
          Append the given unit to the object's list of units.
```

This method should always be used rather than trying to modify the list manually.

Parameters unit (TranslationUnit) - The unit that will be added.

## decode (lines)

decode any non-unicode strings in lines with self.encoding

#### detect\_encoding(text, default\_encodings=None)

Try to detect a file encoding from *text*, using either the chardet lib or by trying to decode the file.

#### encode (lines)

encode any unicode strings in lines in self.encoding

#### fallback\_detection (text)

Simple detection based on BOM in case chardet is not available.

#### findid(id)

find unit with matching id by checking id\_index

## findunit (source)

Find the unit with the given source string.

Return type TranslationUnit or None

#### findunits (source)

Find the units with the given source string.

Return type TranslationUnit or None

## getheaderplural()

Returns the nplural and plural values from the header.

## getids (filename=None)

return a list of unit ids

## getprojectstyle()

Return the project based on information in the header.

### The project is determined in the following sequence:

- 1. Use the 'X-Project-Style' entry in the header.
- 2. Use 'Report-Msgid-Bug-To' entry
- 3. Use the 'X-Accelerator' entry
- 4. Use the Project ID
- 5. Analyse the file itself (not yet implemented)

#### getsourcelanguage()

Get the source language for this store.

## gettargetlanguage()

Return the target language based on information in the header.

## The target language is determined in the following sequence:

- 1. Use the 'Language' entry in the header.
- 2. Poedit's custom headers.
- 3. Analysing the 'Language-Team' entry.

## getunits()

Return a list of all units in this store.

#### header()

Returns the header element, or None. Only the first element is allowed to be a header. Note that this could still return an empty header element, if present.

```
init_headers (charset='UTF-8', encoding='8bit', **kwargs)
sets default values for po headers
```

#### isempty()

Return True if the object doesn't contain any translation units.

## makeheader(\*\*kwargs)

Create a header for the given filename.

Check .makeheaderdict() for information on parameters.

```
makeheaderdict (charset='CHARSET', encoding='ENCODING', project_id_version=None, pot_creation_date=None, po_revision_date=None, last_translator=None, language_team=None, mime_version=None, plural_forms=None, report_msgid_bugs_to=None, **kwargs)
```

Create a header dictionary with useful defaults.

pot\_creation\_date can be None (current date) or a value (datetime or string) po\_revision\_date can be None (form), False (=pot\_creation\_date), True (=now), or a value (datetime or string)

**Returns** Dictionary with the header items

Return type dict of strings

#### makeindex()

Indexes the items in this store. At least .sourceindex should be useful.

#### merge\_on

The matching criterion to use when merging on.

### mergeheaders (otherstore)

Merges another header with this header.

This header is assumed to be the template.

#### parse (input)

Parses the given file or file source string.

## classmethod parsefile(storefile)

Reads the given file (or opens the given filename) and parses back to an object.

## parseheader()

Parses the PO header and returns the interpreted values as a dictionary.

## classmethod parsestring(storestring)

Convert the string representation back to an object.

#### remove\_unit\_from\_index(unit)

Remove a unit from source and locaton indexes

## removeduplicates (duplicatestyle='merge')

Make sure each msgid is unique; merge comments etc from duplicates into original

#### require index()

make sure source index exists

## save()

Save to the file that data was originally read from, if available.

```
savefile (storefile)
          Write the string representation to the given file (or filename).
     serialize(out)
          Write to file
     setprojectstyle (project_style)
          Set the project in the header.
              Parameters project_style (str) - the new project
     setsourcelanguage (sourcelanguage)
          Set the source language for this store.
     settargetlanguage (lang)
          Set the target language in the header.
          This removes any custom Poedit headers if they exist.
              Parameters lang(str) – the new target language code
     translate(source)
          Return the translated string for a given source string.
              Return type String or None
     unit iter()
          Iterator over all the units in this store.
     updatecontributor (name, email=None)
          Add contribution comments if necessary.
     updateheader (add=False, **kwargs)
          Updates the fields in the PO style header.
          This will create a header if add == True.
     updateheaderplural (nplurals, plural)
          Update the Plural-Form PO header.
class translate.storage.pypo.pounit(source=None, wrapper=None, **kwargs)
     adderror (errorname, errortext)
          Adds an error message to this unit.
     addlocation (location)
          Add a location to sourcecomments in the PO unit
              Parameters location (String) – Text location e.g. 'file.c:23' does not include #:
     addlocations (location)
          Add a location or a list of locations.
          Note:
                       Most classes shouldn't need to implement this, but should rather implement
```

**Warning:** This method might be removed in future.

TranslationUnit.addlocation().

```
addnote (text, origin=None, position='append')
     This is modeled on the XLIFF method.
     See translate.storage.xliff.xliffunit.addnote()
classmethod buildfromunit (unit)
     Build a native unit from a foreign unit, preserving as much information as possible.
getalttrans()
     Return a list of alternate units.
     Previous msgid and current msgstr is combined to form a single alternative unit.
getcontext()
     Get the message context.
geterrors()
     Get all error messages.
getid()
     Returns a unique identifier for this unit.
getlocations()
     Get a list of locations from source comments in the PO unit
     rtype: List return: A list of the locations with '#: ' stripped
getnotes (origin=None)
     Return comments based on origin value.
         Parameters origin – programmer, developer, source code, translator or None
gettargetlen()
     Returns the length of the target string.
         Return type Integer
     Note: Plural forms might be combined.
getunits()
     This unit in a list.
hasmarkedcomment (commentmarker)
     Check whether the given comment marker is present.
     These should appear as:
       (commentmarker) ...
hasplural()
     returns whether this pounit contains plural strings...
hastypecomment (typecomment)
     Check whether the given type comment is present
infer_state()
```

Empty method that should be overridden in sub-classes to infer the current state(\_n) of the unit from its

current state.

Used to see if this unit has no source or target string.

isblank()

**Note:** This is probably used more to find translatable units, and we might want to move in that direction rather and get rid of this.

### isfuzzy()

Indicates whether this unit is fuzzy.

#### isheader()

Indicates whether this unit is a header.

#### isobsolete()

indicate whether a unit is obsolete

#### isreview()

Indicates whether this unit needs review.

#### istranslatable()

Indicates whether this unit can be translated.

This should be used to distinguish real units for translation from header, obsolete, binary or other blank units.

#### istranslated()

Indicates whether this unit is translated.

This should be used rather than deducing it from .target, to ensure that other classes can implement more functionality (as XLIFF does).

#### makeobsolete()

Makes this unit obsolete

## markfuzzy (present=True)

Marks the unit as fuzzy or not.

## markreviewneeded (needsreview=True, explanation=None)

Marks the unit to indicate whether it needs review. Adds an optional explanation as a note.

#### **merge** (otherpo, overwrite=False, comments=True, authoritative=False)

Merges the otherpo (with the same msgid) into this one.

Overwrite non-blank self.msgstr only if overwrite is True merge comments only if comments is True

## msgidcomment

Extract KDE style msgid comments from the unit.

## Return type String

**Returns** Returns the extracted msgidcomments found in this unit's msgid.

#### multistring\_to\_rich (mulstring)

Convert a multistring to a list of "rich" string trees:

### prev\_source

Returns the unescaped msgid

#### removenotes (origin=None)

Remove all the translator's notes (other comments)

```
resurrect()
         Makes an obsolete unit normal
     rich_source
          See also:
          rich_to_multistring(), multistring_to_rich()
     rich_target
          See also:
          rich_to_multistring(), multistring_to_rich()
     classmethod rich_to_multistring(elem_list)
          Convert a "rich" string tree to a multistring:
         >>> from translate.storage.placeables.interfaces import X
         >>> rich = [StringElem(['foo', X(id='xxx', sub=[' ']), 'bar'])]
          >>> TranslationUnit.rich_to_multistring(rich)
          multistring(u'foo bar')
     setcontext (context)
          Set the message context
     setid(value)
         Sets the unique identified for this unit.
          only implemented if format allows ids independant from other unit properties like source or context
     settypecomment (typecomment, present=True)
          Alters whether a given typecomment is present
     source
          Returns the unescaped msgid
     target
         Returns the unescaped msgstr
     unit iter()
         Iterator that only returns this unit.
translate.storage.pypo.quoteforpo(text, wrapper_obj=None)
     Quotes the given text for a PO file, returning quoted and escaped lines
translate.storage.pypo.splitlines(text)
     Split lines based on first newline char.
```

Can not use univerzal newlines as they match any newline like character inside text and that breaks on files with unix newlines and LF chars inside comments.

The code looks for first msgid and looks for newline used after it. This should safely cover weird newlines used in comments or filenames, while properly parsing po files with any newlines.

```
translate.storage.pypo.unescape(line)
```

Unescape the given line.

Quotes on either side should already have been removed.

#### qm

Module for parsing Qt .qm files.

**Note:** Based on documentation from Gettext's .qm implementation (see *write-qt.c*) and on observation of the output of Irelease.

**Note:** Certain deprecated section tags are not implemented. These will break and print out the missing tag. They are easy to implement and should follow the structure in 03 (Translation). We could find no examples that use these so we'd rather leave it unimplemented until we actually have test data.

**Note:** Many .qm files are unable to be parsed as they do not have the source text. We assume that since they use a hash table to lookup the data there is actually no need for the source text. It seems however that in Qt4's lrelease all data is included in the resultant .qm file.

**Note:** We can only parse, not create, a .qm file. The main issue is that we need to implement the hashing algorithm (which seems to be identical to the Gettext hash algorithm). Unlike Gettext it seems that the hash is required, but that has not been validated.

**Note:** The code can parse files correctly. But it could be cleaned up to be more readable, especially the part that breaks the file into sections.

http://qt.gitorious.org/+kde-developers/qt/kde-qt/blobs/master/tools/linguist/shared/qm.cpp Plural information QLocale languages

```
class translate.storage.qm.qmfile(inputfile=None, **kwargs)
    A class representing a .qm file.
```

### UnitClass

alias of qmunit

## add\_unit\_to\_index(unit)

Add a unit to source and location idexes

## addsourceunit (source)

Add and returns a new unit with the given source string.

Return type TranslationUnit

## addunit (unit)

Append the given unit to the object's list of units.

This method should always be used rather than trying to modify the list manually.

Parameters unit (TranslationUnit) - The unit that will be added.

#### detect\_encoding (text, default\_encodings=None)

Try to detect a file encoding from *text*, using either the chardet lib or by trying to decode the file.

## fallback\_detection (text)

Simple detection based on BOM in case chardet is not available.

```
findid(id)
     find unit with matching id by checking id_index
findunit (source)
     Find the unit with the given source string.
         Return type TranslationUnit or None
findunits (source)
     Find the units with the given source string.
         Return type TranslationUnit or None
getids (filename=None)
     return a list of unit ids
getprojectstyle()
     Get the project type for this store.
getsourcelanguage()
     Get the source language for this store.
gettargetlanguage()
     Get the target language for this store.
getunits()
     Return a list of all units in this store.
isempty()
     Return True if the object doesn't contain any translation units.
makeindex()
     Indexes the items in this store. At least .sourceindex should be useful.
merge_on
     The matching criterion to use when merging on.
         Returns The default matching criterion for all the subclasses.
         Return type string
parse (input)
     Parses the given file or file source string.
classmethod parsefile (storefile)
     Reads the given file (or opens the given filename) and parses back to an object.
classmethod parsestring(storestring)
     Convert the string representation back to an object.
remove_unit_from_index(unit)
     Remove a unit from source and locaton indexes
require_index()
     make sure source index exists
save()
     Save to the file that data was originally read from, if available.
savefile (storefile)
     Write the string representation to the given file (or filename).
serialize(out)
     Output a string representation of the .qm data file
```

## setprojectstyle (project\_style)

Set the project type for this store.

## setsourcelanguage (sourcelanguage)

Set the source language for this store.

## settargetlanguage (targetlanguage)

Set the target language for this store.

#### translate(source)

Return the translated string for a given source string.

## Return type String or None

#### unit iter()

Iterator over all the units in this store.

## class translate.storage.qm.qmunit(source=None)

A class representing a .qm translation message.

### adderror (errorname, errortext)

Adds an error message to this unit.

#### **Parameters**

- **errorname** (*string*) A single word to id the error.
- **errortext** (*string*) The text describing the error.

#### addlocation (location)

Add one location to the list of locations.

**Note:** Shouldn't be implemented if the format doesn't support it.

### addlocations (location)

Add a location or a list of locations.

**Note:** Most classes shouldn't need to implement this, but should rather implement TranslationUnit.addlocation().

**Warning:** This method might be removed in future.

addnote (text, origin=None, position='append')

Adds a note (comment).

#### **Parameters**

- text (string) Usually just a sentence or two.
- **origin** (string) Specifies who/where the comment comes from. Origin can be one of the following text strings: 'translator' 'developer', 'programmer', 'source code' (synonyms)

## classmethod buildfromunit (unit)

Build a native unit from a foreign unit, preserving as much information as possible.

## getcontext()

Get the message context.

#### geterrors()

Get all error messages.

## **Return type** Dictionary

#### getid()

A unique identifier for this unit.

## Return type string

**Returns** an identifier for this unit that is unique in the store

Derived classes should override this in a way that guarantees a unique identifier for each unit in the store.

#### getlocations()

A list of source code locations.

#### **Return type** List

**Note:** Shouldn't be implemented if the format doesn't support it.

#### getnotes (origin=None)

Returns all notes about this unit.

It will probably be freeform text or something reasonable that can be synthesised by the format. It should not include location comments (see <code>getlocations()</code>).

## gettargetlen()

Returns the length of the target string.

## **Return type** Integer

**Note:** Plural forms might be combined.

## getunits()

This unit in a list.

## hasplural()

Tells whether or not this specific unit has plural strings.

## infer\_state()

Empty method that should be overridden in sub-classes to infer the current state(\_n) of the unit from its current state.

#### isblank()

Used to see if this unit has no source or target string.

**Note:** This is probably used more to find translatable units, and we might want to move in that direction rather and get rid of this.

## isfuzzy()

Indicates whether this unit is fuzzy.

### isheader()

Indicates whether this unit is a header.

## isobsolete()

indicate whether a unit is obsolete

#### isreview()

Indicates whether this unit needs review.

#### istranslatable()

Indicates whether this unit can be translated.

This should be used to distinguish real units for translation from header, obsolete, binary or other blank units.

#### istranslated()

Indicates whether this unit is translated.

This should be used rather than deducing it from .target, to ensure that other classes can implement more functionality (as XLIFF does).

#### makeobsolete()

Make a unit obsolete

#### markfuzzy (value=True)

Marks the unit as fuzzy or not.

### markreviewneeded (needsreview=True, explanation=None)

Marks the unit to indicate whether it needs review.

#### **Parameters**

- needsreview Defaults to True.
- **explanation** Adds an optional explanation as a note.

**merge** (otherunit, overwrite=False, comments=True, authoritative=False)

Do basic format agnostic merging.

## multistring\_to\_rich(mulstring)

Convert a multistring to a list of "rich" string trees:

#### removenotes (origin=None)

Remove all the translator's notes.

#### rich\_source

## See also:

```
rich_to_multistring(), multistring_to_rich()
```

### rich\_target

## See also:

```
rich_to_multistring(), multistring_to_rich()
```

## ${\tt classmethod\ rich\_to\_multistring}\,(\textit{elem\_list})$

Convert a "rich" string tree to a multistring:

```
>>> from translate.storage.placeables.interfaces import X
>>> rich = [StringElem(['foo', X(id='xxx', sub=[' ']), 'bar'])]
>>> TranslationUnit.rich_to_multistring(rich)
multistring(u'foo bar')
```

#### setcontext (context)

Set the message context

#### setid(value)

Sets the unique identified for this unit.

only implemented if format allows ids independant from other unit properties like source or context

## unit\_iter()

Iterator that only returns this unit.

```
translate.storage.qm.qmunpack(file_='messages.qm')
```

Helper to unpack Qt .qm files into a Python string

### qph

Module for handling Qt Linguist Phrase Book (.qph) files.

Extract from the Qt Linguist Manual: Translators: .qph Qt Phrase Book Files are human-readable XML files containing standard phrases and their translations. These files are created and updated by Qt Linguist and may be used by any number of projects and applications.

A DTD to define the format does not seem to exist, but the following code provides the reference implementation for the Qt Linguist product.

Class representing a QPH file store.

#### UnitClass

alias of QphUnit

#### add unit to index (unit)

Add a unit to source and location idexes

## addheader()

Method to be overridden to initialise headers, etc.

## addsourceunit (source)

Adds and returns a new unit with the given string as first entry.

#### addunit (unit, new=True)

Append the given unit to the object's list of units.

This method should always be used rather than trying to modify the list manually.

**Parameters unit** (TranslationUnit) – The unit that will be added.

```
detect_encoding (text, default_encodings=None)
```

Try to detect a file encoding from text, using either the chardet lib or by trying to decode the file.

### fallback\_detection (text)

Simple detection based on BOM in case chardet is not available.

## findid(id)

find unit with matching id by checking id\_index

#### findunit (source)

Find the unit with the given source string.

Return type TranslationUnit or None

#### findunits (source)

Find the units with the given source string.

Return type TranslationUnit or None

## getids (filename=None)

return a list of unit ids

#### getprojectstyle()

Get the project type for this store.

### getsourcelanguage()

Get the source language for this .qph file.

We don't implement setsourcelanguage as users really shouldn't be altering the source language in .qph files, it should be set correctly by the extraction tools.

Returns ISO code e.g. af, fr, pt\_BR

Return type String

## gettargetlanguage()

Get the target language for this .qph file.

Returns ISO code e.g. af, fr, pt\_BR

Return type String

## getunits()

Return a list of all units in this store.

## initbody()

Initialises self.body so it never needs to be retrieved from the XML again.

### isempty()

Return True if the object doesn't contain any translation units.

### makeindex()

Indexes the items in this store. At least .sourceindex should be useful.

## merge\_on

The matching criterion to use when merging on.

**Returns** The default matching criterion for all the subclasses.

Return type string

## namespaced(name)

Returns name in Clark notation.

For example namespaced ("source") in an XLIFF document might return:

{urn:oasis:names:tc:xliff:document:1.1}source

This is needed throughout lxml.

## parse (xml)

Populates this object from the given xml string

## classmethod parsefile(storefile)

Reads the given file (or opens the given filename) and parses back to an object.

# classmethod parsestring(storestring) Convert the string representation back to an object. remove\_unit\_from\_index(unit) Remove a unit from source and locaton indexes require index() make sure source index exists save() Save to the file that data was originally read from, if available. savefile (storefile) Write the string representation to the given file (or filename). serialize(out) Write the XML document to the file *out*. We have to override this to ensure mimic the Qt convention: • no XML declaration setprojectstyle (project\_style) Set the project type for this store. setsourcelanguage (sourcelanguage) Set the source language for this store.

## settargetlanguage (targetlanguage)

Set the target language for this .qph file to targetlanguage.

Parameters targetlanguage (String) – ISO code e.g. af, fr, pt\_BR

## translate(source)

Return the translated string for a given source string.

Return type String or None

## unit\_iter()

Iterator over all the units in this store.

class translate.storage.qph.QphUnit (source, empty=False, \*\*kwargs)
 A single term in the qph file.

## adderror (errorname, errortext)

Adds an error message to this unit.

#### **Parameters**

- **errorname** (*string*) A single word to id the error.
- **errortext** (*string*) The text describing the error.

## addlocation (location)

Add one location to the list of locations.

**Note:** Shouldn't be implemented if the format doesn't support it.

## addlocations (location)

Add a location or a list of locations.

**Note:** Most classes shouldn't need to implement this, but should rather implement TranslationUnit.addlocation().

**Warning:** This method might be removed in future.

### addnote (text, origin=None, position='append')

Add a note specifically in a "definition" tag

#### classmethod buildfromunit (unit)

Build a native unit from a foreign unit, preserving as much information as possible.

### createlanguageNode (lang, text, purpose)

Returns an xml Element setup with given parameters.

## getNodeText (languageNode, xml\_space='preserve')

Retrieves the term from the given languageNode.

## getcontext()

Get the message context.

#### geterrors()

Get all error messages.

## **Return type** Dictionary

#### getid()

A unique identifier for this unit.

### Return type string

**Returns** an identifier for this unit that is unique in the store

Derived classes should override this in a way that guarantees a unique identifier for each unit in the store.

## getlanguageNode (lang=None, index=None)

Retrieves a languageNode either by language or by index.

## getlanguageNodes()

We override this to get source and target nodes.

#### getlocations()

A list of source code locations.

## Return type List

**Note:** Shouldn't be implemented if the format doesn't support it.

### getnotes (origin=None)

Returns all notes about this unit.

It will probably be freeform text or something reasonable that can be synthesised by the format. It should not include location comments (see getlocations()).

## gettarget (lang=None)

retrieves the "target" text (second entry), or the entry in the specified language, if it exists

#### gettargetlen()

Returns the length of the target string.

### **Return type** Integer

Note: Plural forms might be combined.

### getunits()

This unit in a list.

#### hasplural()

Tells whether or not this specific unit has plural strings.

### infer\_state()

Empty method that should be overridden in sub-classes to infer the current state(\_n) of the unit from its current state.

#### isblank()

Used to see if this unit has no source or target string.

**Note:** This is probably used more to find translatable units, and we might want to move in that direction rather and get rid of this.

## isfuzzy()

Indicates whether this unit is fuzzy.

#### isheader()

Indicates whether this unit is a header.

## isobsolete()

indicate whether a unit is obsolete

## isreview()

Indicates whether this unit needs review.

#### istranslatable()

Indicates whether this unit can be translated.

This should be used to distinguish real units for translation from header, obsolete, binary or other blank units.

## istranslated()

Indicates whether this unit is translated.

This should be used rather than deducing it from .target, to ensure that other classes can implement more functionality (as XLIFF does).

### makeobsolete()

Make a unit obsolete

## markfuzzy(value=True)

Marks the unit as fuzzy or not.

## $\verb|markreviewneeded| (needs review=True, explanation=None)$

Marks the unit to indicate whether it needs review.

#### **Parameters**

- needsreview Defaults to True.
- **explanation** Adds an optional explanation as a note.

```
merge (otherunit, overwrite=False, comments=True, authoritative=False)
```

Do basic format agnostic merging.

## multistring\_to\_rich (mulstring)

Convert a multistring to a list of "rich" string trees:

## namespaced(name)

Returns name in Clark notation.

For example namespaced ("source") in an XLIFF document might return:

```
{urn:oasis:names:tc:xliff:document:1.1}source
```

This is needed throughout lxml.

## removenotes (origin=None)

Remove all the translator notes.

```
rich_source
```

#### See also:

```
rich_to_multistring(), multistring_to_rich()
```

## rich\_target

#### See also:

```
rich_to_multistring(), multistring_to_rich()
```

## classmethod rich\_to\_multistring(elem\_list)

Convert a "rich" string tree to a multistring:

```
>>> from translate.storage.placeables.interfaces import X
>>> rich = [StringElem(['foo', X(id='xxx', sub=[' ']), 'bar'])]
>>> TranslationUnit.rich_to_multistring(rich)
multistring(u'foo bar')
```

## setcontext (context)

Set the message context

#### setid(value)

Sets the unique identified for this unit.

only implemented if format allows ids independant from other unit properties like source or context

```
settarget (target, lang='xx', append=False)
```

Sets the "target" string (second language), or alternatively appends to the list

#### unit iter()

Iterator that only returns this unit.

```
rc
```

```
Classes that hold units of .rc files (rcunit) or entire files (rcfile) used in translating Windows Resources.
translate.storage.rc.escape_to_python(string)
     Escape a given .rc string into a valid Python string.
translate.storage.rc.escape_to_rc(string)
     Escape a given Python string into a valid .rc string.
class translate.storage.rc.rcfile(inputfile=None, lang=None, sublang=None, **kwargs)
     This class represents a .rc file, made up of rcunits.
     UnitClass
          alias of rcunit
     add unit to index (unit)
          Add a unit to source and location idexes
     addsourceunit (source)
          Add and returns a new unit with the given source string.
               Return type TranslationUnit
     addunit (unit)
          Append the given unit to the object's list of units.
          This method should always be used rather than trying to modify the list manually.
               Parameters unit (TranslationUnit) – The unit that will be added.
     detect encoding (text, default encodings=None)
          Try to detect a file encoding from text, using either the chardet lib or by trying to decode the file.
     fallback_detection (text)
          Simple detection based on BOM in case chardet is not available.
     findid(id)
          find unit with matching id by checking id_index
     findunit (source)
          Find the unit with the given source string.
               Return type TranslationUnit or None
     findunits (source)
          Find the units with the given source string.
              Return type TranslationUnit or None
     getids (filename=None)
          return a list of unit ids
     getprojectstyle()
          Get the project type for this store.
     getsourcelanguage()
          Get the source language for this store.
     gettargetlanguage()
          Get the target language for this store.
     getunits()
```

Return a list of all units in this store.

```
isempty()
           Return True if the object doesn't contain any translation units.
     makeindex()
           Indexes the items in this store. At least .sourceindex should be useful.
     merge_on
           The matching criterion to use when merging on.
               Returns The default matching criterion for all the subclasses.
               Return type string
     parse (rcsrc)
           Read the source of a .rc file in and include them as units.
     classmethod parsefile(storefile)
           Reads the given file (or opens the given filename) and parses back to an object.
     classmethod parsestring(storestring)
           Convert the string representation back to an object.
     remove_unit_from_index(unit)
           Remove a unit from source and locaton indexes
     require_index()
           make sure source index exists
           Save to the file that data was originally read from, if available.
     savefile (storefile)
           Write the string representation to the given file (or filename).
     serialize(out)
           Write the units back to file.
     setprojectstyle (project_style)
           Set the project type for this store.
     setsourcelanguage (sourcelanguage)
           Set the source language for this store.
     settargetlanguage (targetlanguage)
           Set the target language for this store.
     translate(source)
           Return the translated string for a given source string.
               Return type String or None
     unit iter()
           Iterator over all the units in this store.
class translate.storage.rc.rcunit(source=", **kwargs)
     A unit of an rc file
     adderror (errorname, errortext)
           Adds an error message to this unit.
               Parameters
                   • errorname (string) – A single word to id the error.
```

4.1. API 607

• **errortext** (*string*) – The text describing the error.

#### addlocation (location)

Add one location to the list of locations.

**Note:** Shouldn't be implemented if the format doesn't support it.

#### addlocations (location)

Add a location or a list of locations.

**Note:** Most classes shouldn't need to implement this, but should rather implement TranslationUnit.addlocation().

**Warning:** This method might be removed in future.

## addnote (text, origin=None, position='append')

Adds a note (comment).

#### **Parameters**

- **text** (*string*) Usually just a sentence or two.
- **origin** (string) Specifies who/where the comment comes from. Origin can be one of the following text strings: 'translator' 'developer', 'programmer', 'source code' (synonyms)

### classmethod buildfromunit (unit)

Build a native unit from a foreign unit, preserving as much information as possible.

#### getcontext()

Get the message context.

## geterrors()

Get all error messages.

Return type Dictionary

## getid()

A unique identifier for this unit.

## Return type string

Returns an identifier for this unit that is unique in the store

Derived classes should override this in a way that guarantees a unique identifier for each unit in the store.

### getlocations()

A list of source code locations.

### **Return type** List

**Note:** Shouldn't be implemented if the format doesn't support it.

## getnotes (origin=None)

Returns all notes about this unit.

It will probably be freeform text or something reasonable that can be synthesised by the format. It should not include location comments (see <code>getlocations()</code>).

### getoutput()

Convert the element back into formatted lines for a .rc file.

### gettargetlen()

Returns the length of the target string.

### **Return type** Integer

Note: Plural forms might be combined.

### getunits()

This unit in a list.

## hasplural()

Tells whether or not this specific unit has plural strings.

### infer\_state()

Empty method that should be overridden in sub-classes to infer the current state(\_n) of the unit from its current state.

#### isblank()

Returns whether this is a blank element, containing only comments.

# isfuzzy()

Indicates whether this unit is fuzzy.

### isheader()

Indicates whether this unit is a header.

### isobsolete()

indicate whether a unit is obsolete

# isreview()

Indicates whether this unit needs review.

### istranslatable()

Indicates whether this unit can be translated.

This should be used to distinguish real units for translation from header, obsolete, binary or other blank units.

## istranslated()

Indicates whether this unit is translated.

This should be used rather than deducing it from .target, to ensure that other classes can implement more functionality (as XLIFF does).

#### makeobsolete()

Make a unit obsolete

# markfuzzy(value=True)

Marks the unit as fuzzy or not.

### markreviewneeded (needsreview=True, explanation=None)

Marks the unit to indicate whether it needs review.

## **Parameters**

- needsreview Defaults to True.
- **explanation** Adds an optional explanation as a note.

```
\textbf{merge} \ (other unit, overwrite = False, comments = True, authoritative = False)
```

Do basic format agnostic merging.

```
multistring_to_rich (mulstring)
```

Convert a multistring to a list of "rich" string trees:

### removenotes (origin=None)

Remove all the translator's notes.

#### rich source

### See also:

```
rich_to_multistring(), multistring_to_rich()
```

#### rich\_target

### See also:

```
rich_to_multistring(), multistring_to_rich()
```

### classmethod rich\_to\_multistring(elem\_list)

Convert a "rich" string tree to a multistring:

```
>>> from translate.storage.placeables.interfaces import X
>>> rich = [StringElem(['foo', X(id='xxx', sub=[' ']), 'bar'])]
>>> TranslationUnit.rich_to_multistring(rich)
multistring(u'foo bar')
```

## setcontext (context)

Set the message context

### setid(value)

Sets the unique identified for this unit.

only implemented if format allows ids independent from other unit properties like source or context

## unit\_iter()

Iterator that only returns this unit.

## statistics

Module to provide statistics and related functionality.

```
{\bf class} \ {\bf translate.storage.statistics.Statistics} \ (source language='en', \ target-language='en', checkers tyle=None)
```

Manages statistics for storage objects.

# classifyunit (unit)

Returns a list of the classes that the unit belongs to.

**Parameters** unit – the unit to classify

### classifyunits()

Makes a dictionary of which units fall into which classifications.

This method iterates over all units.

### countwords()

Counts the source and target words in each of the units.

### fuzzy\_unitcount()

Returns the number of fuzzy units.

### fuzzy\_units()

Return a list of fuzzy units.

### get\_source\_text (units)

Joins the unit source strings in a single string of text.

### getunits()

Returns a list of all units in this object.

### reclassifyunit (item)

Updates the classification of a unit in self.classification.

**Parameters** item – an integer that is an index in .getunits().

# source\_wordcount()

Returns the number of words in the source text.

### translated unitcount()

Returns the number of translated units.

### translated\_units()

Return a list of translated units.

## translated\_wordcount()

Returns the number of translated words in this object.

### untranslated\_unitcount()

Returns the number of untranslated units.

### untranslated\_units()

Return a list of untranslated units.

## untranslated wordcount()

Returns the number of untranslated words in this object.

### wordcount (text)

Returns the number of words in the given text.

### statsdb

Module to provide a cache of statistics in a database.

### class translate.storage.statsdb.StatsCache

An object instantiated as a singleton for each statsfile that provides access to the database cache from a pool of StatsCache objects.

### con = None

This cache's connection

### cur = None

The current cursor

```
filechecks (filename, checker, store=None)
     Retrieves the error statistics for the given file if possible, otherwise delegates to cachestorechecks().
filestatestats (filename, store=None, extended=False)
```

Return a dictionary of unit stats mapping sets of unit indices with those states

**filestats** (filename, checker, store=None, extended=False)

Return a dictionary of property names mapping sets of unit indices with those properties.

```
filetotals (filename, store=None, extended=False)
```

Retrieves the statistics for the given file if possible, otherwise delegates to cachestore().

```
unitstats (filename, _lang=None, store=None)
```

Return a dictionary of property names mapping to arrays which map unit indices to property values.

Please note that this is different from filestats, since filestats supplies sets of unit indices with a given property, whereas this method supplies arrays which map unit indices to given values.

```
translate.storage.statsdb.emptyfiletotals()
     Returns a dictionary with all statistics initalised to 0.
```

```
translate.storage.statsdb.statefordb(unit)
```

Returns the numeric database state for the unit.

```
translate.storage.statsdb.transaction(f)
```

Modifies f to commit database changes if it executes without exceptions. Otherwise it rolls back the database.

ALL publicly accessible methods in StatsCache MUST be decorated with this decorator.

```
translate.storage.statsdb.wordsinunit(unit)
```

Counts the words in the unit's source and target, taking plurals into account. The target words are only counted if the unit is translated.

### subtitles

Class that manages subtitle files for translation.

This class makes use of the subtitle functionality of gaupol.

## See also:

```
gaupol/agents/open.py::open_main
```

A patch to gaupol is required to open utf-8 files successfully.

```
class translate.storage.subtitles.AdvSubStationAlphaFile(*args, **kwargs)
    specialized class for SubRipFile's only
```

### UnitClass

```
alias of SubtitleUnit
```

# add\_unit\_to\_index(unit)

Add a unit to source and location idexes

#### addsourceunit (source)

Add and returns a new unit with the given source string.

```
Return type TranslationUnit
```

```
addunit (unit)
```

Append the given unit to the object's list of units.

This method should always be used rather than trying to modify the list manually.

```
detect_encoding (text, default_encodings=None)
     Try to detect a file encoding from text, using either the chardet lib or by trying to decode the file.
fallback detection (text)
     Simple detection based on BOM in case chardet is not available.
findid(id)
     find unit with matching id by checking id_index
findunit (source)
     Find the unit with the given source string.
         Return type TranslationUnit or None
findunits (source)
     Find the units with the given source string.
         Return type TranslationUnit or None
getids (filename=None)
     return a list of unit ids
getprojectstyle()
     Get the project type for this store.
getsourcelanguage()
     Get the source language for this store.
gettargetlanguage()
     Get the target language for this store.
getunits()
     Return a list of all units in this store.
isempty()
     Return True if the object doesn't contain any translation units.
makeindex()
     Indexes the items in this store. At least .sourceindex should be useful.
merge on
     The matching criterion to use when merging on.
         Returns The default matching criterion for all the subclasses.
         Return type string
parse (input)
     parser to process the given source string
classmethod parsefile(storefile)
     parse the given file
classmethod parsestring(storestring)
     Convert the string representation back to an object.
remove unit from index(unit)
     Remove a unit from source and locaton indexes
require_index()
     make sure source index exists
```

**Parameters unit** (TranslationUnit) – The unit that will be added.

```
save()
           Save to the file that data was originally read from, if available.
     savefile (storefile)
           Write the string representation to the given file (or filename).
     serialize (out)
           Converts to a bytes representation that can be parsed back using parsestring (). out should be an
           open file-like objects to write to.
     setprojectstyle (project_style)
           Set the project type for this store.
     setsourcelanguage (sourcelanguage)
           Set the source language for this store.
     settargetlanguage (targetlanguage)
           Set the target language for this store.
     translate(source)
           Return the translated string for a given source string.
               Return type String or None
     unit_iter()
          Iterator over all the units in this store.
class translate.storage.subtitles.MicroDVDFile(*args, **kwargs)
     specialized class for SubRipFile's only
     UnitClass
           alias of SubtitleUnit
     add_unit_to_index(unit)
           Add a unit to source and location idexes
     addsourceunit (source)
           Add and returns a new unit with the given source string.
               Return type TranslationUnit
     addunit (unit)
           Append the given unit to the object's list of units.
           This method should always be used rather than trying to modify the list manually.
               Parameters unit (TranslationUnit) - The unit that will be added.
     detect encoding(text, default encodings=None)
           Try to detect a file encoding from text, using either the chardet lib or by trying to decode the file.
     fallback detection(text)
           Simple detection based on BOM in case chardet is not available.
     findid(id)
           find unit with matching id by checking id_index
     findunit (source)
           Find the unit with the given source string.
               Return type TranslationUnit or None
     findunits (source)
           Find the units with the given source string.
```

# Return type TranslationUnit or None getids (filename=None) return a list of unit ids getprojectstyle() Get the project type for this store. getsourcelanguage() Get the source language for this store. gettargetlanguage() Get the target language for this store. getunits() Return a list of all units in this store. isempty() Return True if the object doesn't contain any translation units. makeindex() Indexes the items in this store. At least .sourceindex should be useful. merge on The matching criterion to use when merging on. **Returns** The default matching criterion for all the subclasses. Return type string parse (input) parser to process the given source string classmethod parsefile(storefile) parse the given file classmethod parsestring(storestring) Convert the string representation back to an object. remove\_unit\_from\_index(unit) Remove a unit from source and locaton indexes require index() make sure source index exists save() Save to the file that data was originally read from, if available. savefile (storefile) Write the string representation to the given file (or filename). serialize (out) Converts to a bytes representation that can be parsed back using parsestring(). out should be an open file-like objects to write to. setprojectstyle (project\_style) Set the project type for this store. setsourcelanguage (sourcelanguage) Set the source language for this store. settargetlanguage (targetlanguage)

4.1. API 615

Set the target language for this store.

```
translate(source)
          Return the translated string for a given source string.
               Return type String or None
     unit iter()
          Iterator over all the units in this store.
class translate.storage.subtitles.SubRipFile(*args, **kwargs)
     specialized class for SubRipFile's only
     UnitClass
          alias of SubtitleUnit
     add_unit_to_index(unit)
          Add a unit to source and location idexes
     addsourceunit (source)
          Add and returns a new unit with the given source string.
               Return type TranslationUnit
     addunit (unit)
          Append the given unit to the object's list of units.
          This method should always be used rather than trying to modify the list manually.
               Parameters unit (TranslationUnit) - The unit that will be added.
     detect_encoding(text, default_encodings=None)
          Try to detect a file encoding from text, using either the chardet lib or by trying to decode the file.
     fallback_detection (text)
          Simple detection based on BOM in case chardet is not available.
     findid(id)
          find unit with matching id by checking id_index
     findunit (source)
          Find the unit with the given source string.
               Return type TranslationUnit or None
     findunits (source)
          Find the units with the given source string.
               Return type TranslationUnit or None
     getids (filename=None)
          return a list of unit ids
     getprojectstyle()
          Get the project type for this store.
     getsourcelanguage()
          Get the source language for this store.
     gettargetlanguage()
          Get the target language for this store.
     getunits()
          Return a list of all units in this store.
     isempty()
```

Return True if the object doesn't contain any translation units.

```
makeindex()
          Indexes the items in this store. At least .sourceindex should be useful.
     merge_on
          The matching criterion to use when merging on.
               Returns The default matching criterion for all the subclasses.
               Return type string
     parse(input)
          parser to process the given source string
     classmethod parsefile(storefile)
          parse the given file
     classmethod parsestring(storestring)
          Convert the string representation back to an object.
     remove_unit_from_index(unit)
          Remove a unit from source and locaton indexes
     require_index()
          make sure source index exists
     save()
          Save to the file that data was originally read from, if available.
     savefile (storefile)
          Write the string representation to the given file (or filename).
     serialize(out)
          Converts to a bytes representation that can be parsed back using parsestring(). out should be an
          open file-like objects to write to.
     setprojectstyle (project_style)
          Set the project type for this store.
     setsourcelanguage (sourcelanguage)
          Set the source language for this store.
     settargetlanguage (targetlanguage)
          Set the target language for this store.
     translate(source)
          Return the translated string for a given source string.
               Return type String or None
     unit iter()
          Iterator over all the units in this store.
class translate.storage.subtitles.SubStationAlphaFile(*args, **kwargs)
     specialized class for SubRipFile's only
     UnitClass
          alias of SubtitleUnit
     add_unit_to_index (unit)
          Add a unit to source and location idexes
     addsourceunit (source)
          Add and returns a new unit with the given source string.
               Return type TranslationUnit
```

```
addunit (unit)
     Append the given unit to the object's list of units.
     This method should always be used rather than trying to modify the list manually.
         Parameters unit (TranslationUnit) – The unit that will be added.
detect encoding (text, default encodings=None)
     Try to detect a file encoding from text, using either the chardet lib or by trying to decode the file.
fallback_detection (text)
     Simple detection based on BOM in case chardet is not available.
findid(id)
     find unit with matching id by checking id_index
findunit (source)
     Find the unit with the given source string.
         Return type TranslationUnit or None
findunits (source)
     Find the units with the given source string.
         Return type TranslationUnit or None
getids (filename=None)
     return a list of unit ids
getprojectstyle()
     Get the project type for this store.
getsourcelanguage()
     Get the source language for this store.
gettargetlanguage()
     Get the target language for this store.
getunits()
     Return a list of all units in this store.
isempty()
     Return True if the object doesn't contain any translation units.
makeindex()
     Indexes the items in this store. At least .sourceindex should be useful.
merge_on
     The matching criterion to use when merging on.
         Returns The default matching criterion for all the subclasses.
         Return type string
parse (input)
     parser to process the given source string
classmethod parsefile (storefile)
     parse the given file
classmethod parsestring(storestring)
```

Convert the string representation back to an object.

Remove a unit from source and locaton indexes

 ${\tt remove\_unit\_from\_index}~(\textit{unit})$ 

```
require index()
           make sure source index exists
     save()
           Save to the file that data was originally read from, if available.
     savefile (storefile)
           Write the string representation to the given file (or filename).
           Converts to a bytes representation that can be parsed back using parsestring(). out should be an
           open file-like objects to write to.
     setprojectstyle (project_style)
           Set the project type for this store.
     setsourcelanguage (sourcelanguage)
           Set the source language for this store.
     settargetlanguage (targetlanguage)
           Set the target language for this store.
     translate(source)
           Return the translated string for a given source string.
               Return type String or None
     unit iter()
           Iterator over all the units in this store.
class translate.storage.subtitles.SubtitleFile(inputfile=None, **kwargs)
     A subtitle file
     UnitClass
          alias of SubtitleUnit
     add_unit_to_index(unit)
           Add a unit to source and location idexes
     addsourceunit (source)
           Add and returns a new unit with the given source string.
               Return type TranslationUnit
     addunit (unit)
           Append the given unit to the object's list of units.
           This method should always be used rather than trying to modify the list manually.
               Parameters unit (TranslationUnit) - The unit that will be added.
     detect_encoding(text, default_encodings=None)
           Try to detect a file encoding from text, using either the chardet lib or by trying to decode the file.
     fallback_detection (text)
           Simple detection based on BOM in case chardet is not available.
     findid(id)
           find unit with matching id by checking id_index
     findunit (source)
           Find the unit with the given source string.
               Return type TranslationUnit or None
```

```
findunits (source)
     Find the units with the given source string.
         Return type TranslationUnit or None
getids (filename=None)
     return a list of unit ids
getprojectstyle()
     Get the project type for this store.
getsourcelanguage()
     Get the source language for this store.
gettargetlanguage()
     Get the target language for this store.
getunits()
     Return a list of all units in this store.
isempty()
     Return True if the object doesn't contain any translation units.
makeindex()
     Indexes the items in this store. At least .sourceindex should be useful.
merge_on
     The matching criterion to use when merging on.
         Returns The default matching criterion for all the subclasses.
         Return type string
parse(input)
     parser to process the given source string
classmethod parsefile(storefile)
     parse the given file
classmethod parsestring(storestring)
     Convert the string representation back to an object.
remove_unit_from_index(unit)
     Remove a unit from source and locaton indexes
require_index()
     make sure source index exists
save()
     Save to the file that data was originally read from, if available.
savefile (storefile)
     Write the string representation to the given file (or filename).
serialize (out)
     Converts to a bytes representation that can be parsed back using parsestring (). out should be an
     open file-like objects to write to.
setprojectstyle (project_style)
     Set the project type for this store.
setsourcelanguage (sourcelanguage)
     Set the source language for this store.
```

### settargetlanguage (targetlanguage)

Set the target language for this store.

### translate(source)

Return the translated string for a given source string.

## Return type String or None

### unit\_iter()

Iterator over all the units in this store.

### class translate.storage.subtitles.SubtitleUnit(source=None, \*\*kwargs)

A subtitle entry that is translatable

#### adderror (errorname, errortext)

Adds an error message to this unit.

### **Parameters**

- **errorname** (*string*) A single word to id the error.
- **errortext** (*string*) The text describing the error.

### addlocation (location)

Add one location to the list of locations.

**Note:** Shouldn't be implemented if the format doesn't support it.

#### addlocations (location)

Add a location or a list of locations.

**Note:** Most classes shouldn't need to implement this, but should rather implement TranslationUnit.addlocation().

Warning: This method might be removed in future.

### addnote (text, origin=None, position='append')

Adds a note (comment).

# **Parameters**

- **text** (*string*) Usually just a sentence or two.
- **origin** (*string*) Specifies who/where the comment comes from. Origin can be one of the following text strings: 'translator' 'developer', 'programmer', 'source code' (synonyms)

### classmethod buildfromunit (unit)

Build a native unit from a foreign unit, preserving as much information as possible.

# getcontext()

Get the message context.

### geterrors()

Get all error messages.

Return type Dictionary

### getid()

A unique identifier for this unit.

### Return type string

Returns an identifier for this unit that is unique in the store

Derived classes should override this in a way that guarantees a unique identifier for each unit in the store.

#### getlocations()

A list of source code locations.

### **Return type** List

Note: Shouldn't be implemented if the format doesn't support it.

### getnotes (origin=None)

Returns all notes about this unit.

It will probably be freeform text or something reasonable that can be synthesised by the format. It should not include location comments (see getlocations()).

## gettargetlen()

Returns the length of the target string.

**Return type** Integer

Note: Plural forms might be combined.

### getunits()

This unit in a list.

## hasplural()

Tells whether or not this specific unit has plural strings.

#### infer state()

Empty method that should be overridden in sub-classes to infer the current state(\_n) of the unit from its current state.

## isblank()

Used to see if this unit has no source or target string.

**Note:** This is probably used more to find translatable units, and we might want to move in that direction rather and get rid of this.

# isfuzzy()

Indicates whether this unit is fuzzy.

### isheader()

Indicates whether this unit is a header.

#### isobsolete()

indicate whether a unit is obsolete

# isreview()

Indicates whether this unit needs review.

#### istranslatable()

Indicates whether this unit can be translated.

This should be used to distinguish real units for translation from header, obsolete, binary or other blank units.

# istranslated()

Indicates whether this unit is translated.

This should be used rather than deducing it from .target, to ensure that other classes can implement more functionality (as XLIFF does).

### makeobsolete()

Make a unit obsolete

### markfuzzy(value=True)

Marks the unit as fuzzy or not.

### markreviewneeded (needsreview=True, explanation=None)

Marks the unit to indicate whether it needs review.

#### **Parameters**

- needsreview Defaults to True.
- **explanation** Adds an optional explanation as a note.

merge (otherunit, overwrite=False, comments=True, authoritative=False)

Do basic format agnostic merging.

#### multistring to rich(mulstring)

Convert a multistring to a list of "rich" string trees:

# removenotes (origin=None)

Remove all the translator's notes.

#### rich source

## See also:

```
rich_to_multistring(), multistring_to_rich()
```

# rich\_target

#### See also:

```
rich_to_multistring(), multistring_to_rich()
```

## classmethod rich\_to\_multistring(elem\_list)

Convert a "rich" string tree to a multistring:

```
>>> from translate.storage.placeables.interfaces import X
>>> rich = [StringElem(['foo', X(id='xxx', sub=[' ']), 'bar'])]
>>> TranslationUnit.rich_to_multistring(rich)
multistring(u'foo bar')
```

setcontext (context)

```
Set the message context
     setid(value)
           Sets the unique identified for this unit.
           only implemented if format allows ids independant from other unit properties like source or context
     unit_iter()
           Iterator that only returns this unit.
symbian
tbx
module for handling TBX glossary files
class translate.storage.tbx.tbxfile(inputfile=None,
                                                                      sourcelanguage='en',
                                                                                                 target-
                                                  language=None, **kwargs)
     Class representing a TBX file store.
     UnitClass
          alias of tbxunit
     add_unit_to_index(unit)
           Add a unit to source and location idexes
     addheader()
           Initialise headers with TBX specific things.
     addsourceunit (source)
           Adds and returns a new unit with the given string as first entry.
     addunit (unit, new=True)
           Append the given unit to the object's list of units.
           This method should always be used rather than trying to modify the list manually.
               Parameters unit (TranslationUnit) - The unit that will be added.
     detect_encoding (text, default_encodings=None)
           Try to detect a file encoding from text, using either the chardet lib or by trying to decode the file.
     fallback_detection (text)
           Simple detection based on BOM in case chardet is not available.
     findid(id)
           find unit with matching id by checking id_index
     findunit (source)
           Find the unit with the given source string.
               Return type TranslationUnit or None
     findunits (source)
           Find the units with the given source string.
               Return type TranslationUnit or None
     getids (filename=None)
           return a list of unit ids
```

### getprojectstyle()

Get the project type for this store.

### getsourcelanguage()

Get the source language for this store.

## gettargetlanguage()

Get the target language for this store.

#### getunits()

Return a list of all units in this store.

### initbody()

Initialises self.body so it never needs to be retrieved from the XML again.

### isempty()

Return True if the object doesn't contain any translation units.

### makeindex()

Indexes the items in this store. At least .sourceindex should be useful.

#### merge on

The matching criterion to use when merging on.

**Returns** The default matching criterion for all the subclasses.

Return type string

### namespaced(name)

Returns name in Clark notation.

For example namespaced ("source") in an XLIFF document might return:

```
{urn:oasis:names:tc:xliff:document:1.1}source
```

This is needed throughout lxml.

### parse(xml)

Populates this object from the given xml string

### classmethod parsefile(storefile)

Reads the given file (or opens the given filename) and parses back to an object.

# classmethod parsestring(storestring)

Convert the string representation back to an object.

### remove\_unit\_from\_index(unit)

Remove a unit from source and locaton indexes

### require\_index()

make sure source index exists

### save()

Save to the file that data was originally read from, if available.

# ${\tt savefile} \ (storefile)$

Write the string representation to the given file (or filename).

## serialize(out=None)

Converts to a string containing the file's XML

## setprojectstyle (project\_style)

Set the project type for this store.

### setsourcelanguage (sourcelanguage)

Set the source language for this store.

# settargetlanguage (targetlanguage)

Set the target language for this store.

#### translate(source)

Return the translated string for a given source string.

**Return type** String or None

### unit\_iter()

Iterator over all the units in this store.

### **class** translate.storage.tbx.**tbxunit**(source, empty=False, \*\*kwargs)

A single term in the TBX file. Provisional work is done to make several languages possible.

### adderror (errorname, errortext)

Adds an error message to this unit.

#### **Parameters**

- **errorname** (*string*) A single word to id the error.
- **errortext** (*string*) The text describing the error.

### addlocation (location)

Add one location to the list of locations.

**Note:** Shouldn't be implemented if the format doesn't support it.

# addlocations (location)

Add a location or a list of locations.

**Note:** Most classes shouldn't need to implement this, but should rather implement TranslationUnit.addlocation().

**Warning:** This method might be removed in future.

## addnote (text, origin=None, position='append')

Add a note specifically in a "note" tag

# ${\tt classmethod\ buildfromunit}\ (unit)$

Build a native unit from a foreign unit, preserving as much information as possible.

### createlanguageNode (lang, text, purpose)

returns a langset xml Element setup with given parameters

# getNodeText (languageNode, xml\_space='preserve')

Retrieves the term from the given languageNode.

# getcontext()

Get the message context.

# geterrors()

Get all error messages.

**Return type** Dictionary

### getid()

A unique identifier for this unit.

### Return type string

Returns an identifier for this unit that is unique in the store

Derived classes should override this in a way that guarantees a unique identifier for each unit in the store.

### getlanguageNode (lang=None, index=None)

Retrieves a languageNode either by language or by index.

### getlanguageNodes()

Returns a list of all nodes that contain per language information.

### getlocations()

A list of source code locations.

### **Return type** List

**Note:** Shouldn't be implemented if the format doesn't support it.

## getnotes (origin=None)

Returns all notes about this unit.

It will probably be freeform text or something reasonable that can be synthesised by the format. It should not include location comments (see getlocations()).

#### gettarget (lang=None)

retrieves the "target" text (second entry), or the entry in the specified language, if it exists

### gettargetlen()

Returns the length of the target string.

# Return type Integer

Note: Plural forms might be combined.

# getunits()

This unit in a list.

# hasplural()

Tells whether or not this specific unit has plural strings.

#### infer state()

Empty method that should be overridden in sub-classes to infer the current state(\_n) of the unit from its current state.

### isblank()

Used to see if this unit has no source or target string.

**Note:** This is probably used more to find translatable units, and we might want to move in that direction rather and get rid of this.

### isfuzzy()

Indicates whether this unit is fuzzy.

#### isheader()

Indicates whether this unit is a header.

### isobsolete()

indicate whether a unit is obsolete

### isreview()

Indicates whether this unit needs review.

#### istranslatable()

Indicates whether this unit can be translated.

This should be used to distinguish real units for translation from header, obsolete, binary or other blank units.

#### istranslated()

Indicates whether this unit is translated.

This should be used rather than deducing it from .target, to ensure that other classes can implement more functionality (as XLIFF does).

#### makeobsolete()

Make a unit obsolete

# markfuzzy(value=True)

Marks the unit as fuzzy or not.

### markreviewneeded (needsreview=True, explanation=None)

Marks the unit to indicate whether it needs review.

#### **Parameters**

- needsreview Defaults to True.
- **explanation** Adds an optional explanation as a note.

 $\textbf{merge} \ (other unit, overwrite = False, comments = True, authoritative = False)$ 

Do basic format agnostic merging.

### multistring\_to\_rich(mulstring)

Convert a multistring to a list of "rich" string trees:

## namespaced(name)

Returns name in Clark notation.

For example namespaced ("source") in an XLIFF document might return:

```
{urn:oasis:names:tc:xliff:document:1.1}source
```

This is needed throughout lxml.

## removenotes (origin=None)

Remove all the translator notes.

#### rich source

See also:

```
rich_to_multistring(), multistring_to_rich()
rich_target
    See also:
    rich to multistring(), multistring to rich()
classmethod rich_to_multistring(elem_list)
    Convert a "rich" string tree to a multistring:
    >>> from translate.storage.placeables.interfaces import X
    >>> rich = [StringElem(['foo', X(id='xxx', sub=[' ']), 'bar'])]
    >>> TranslationUnit.rich_to_multistring(rich)
    multistring(u'foo bar')
setcontext (context)
    Set the message context
setid(value)
    Sets the unique identified for this unit.
    only implemented if format allows ids independant from other unit properties like source or context
settarget (target, lang='xx', append=False)
    Sets the "target" string (second language), or alternatively appends to the list
```

### tiki

unit\_iter()

Iterator that only returns this unit.

Class that manages TikiWiki files for translation. Tiki files are <strike>ugly and inconsistent</strike> formatted as a single large PHP array with several special sections identified by comments. Example current as of 2008-12-01:

```
</php
// Many comments at the top
$lang=Array(
// ### Start of unused words
"aaa" => "zzz",
// ### end of unused words

// ### start of untranslated words
// "bbb" => "yyy",
// ### end of untranslated words

// ### end of untranslated words

// ### start of possibly untranslated words

"ccc" => "xxx",
// ### end of possibly untranslated words

"ddd" => "www",
"###end###"=>"###end###");
?>
```

In addition there are several auto-generated //-style comments scattered through the page and array, some of which matter when being parsed.

This has all been gleaned from the TikiWiki source. As far as I know no detailed documentation exists for the tiki language.php files.

```
class translate.storage.tiki.TikiStore(inputfile=None)
     Represents a tiki language.php file.
     UnitClass
          alias of TikiUnit
     add unit to index (unit)
          Add a unit to source and location idexes
     addsourceunit (source)
          Add and returns a new unit with the given source string.
               Return type TranslationUnit
     addunit (unit)
          Append the given unit to the object's list of units.
          This method should always be used rather than trying to modify the list manually.
               Parameters unit (TranslationUnit) - The unit that will be added.
     detect encoding(text, default encodings=None)
          Try to detect a file encoding from text, using either the chardet lib or by trying to decode the file.
     fallback_detection (text)
          Simple detection based on BOM in case chardet is not available.
     findid(id)
          find unit with matching id by checking id index
     findunit (source)
          Find the unit with the given source string.
               Return type TranslationUnit or None
     findunits (source)
          Find the units with the given source string.
               Return type TranslationUnit or None
     getids (filename=None)
          return a list of unit ids
     getprojectstyle()
          Get the project type for this store.
     getsourcelanguage()
          Get the source language for this store.
     gettargetlanguage()
          Get the target language for this store.
     getunits()
          Return a list of all units in this store.
     isempty()
          Return True if the object doesn't contain any translation units.
          Indexes the items in this store. At least .sourceindex should be useful.
     merge_on
          The matching criterion to use when merging on.
```

**Returns** The default matching criterion for all the subclasses.

## Return type string

### parse (input)

Parse the given input into source units.

Parameters input – the source, either a string or filehandle

#### classmethod parsefile (storefile)

Reads the given file (or opens the given filename) and parses back to an object.

### classmethod parsestring(storestring)

Convert the string representation back to an object.

### remove\_unit\_from\_index(unit)

Remove a unit from source and locaton indexes

#### require\_index()

make sure source index exists

#### save()

Save to the file that data was originally read from, if available.

### savefile (storefile)

Write the string representation to the given file (or filename).

### serialize(out)

Will return a formatted tiki-style language.php file.

### setprojectstyle (project\_style)

Set the project type for this store.

# setsourcelanguage (sourcelanguage)

Set the source language for this store.

## settargetlanguage (targetlanguage)

Set the target language for this store.

#### translate(source)

Return the translated string for a given source string.

# Return type String or None

# unit\_iter()

Iterator over all the units in this store.

# class translate.storage.tiki.TikiUnit (source=None, \*\*kwargs)

A tiki unit entry.

# adderror(errorname, errortext)

Adds an error message to this unit.

## **Parameters**

- **errorname** (*string*) A single word to id the error.
- **errortext** (*string*) The text describing the error.

### addlocation (location)

Location is defined by the comments in the file. This function will only set valid locations.

**Parameters** location – Where the string is located in the file. Must be a valid location.

## addlocations (location)

Add a location or a list of locations.

**Note:** Most classes shouldn't need to implement this, but should rather implement TranslationUnit.addlocation().

**Warning:** This method might be removed in future.

### addnote (text, origin=None, position='append')

Adds a note (comment).

### **Parameters**

- **text** (*string*) Usually just a sentence or two.
- **origin** (*string*) Specifies who/where the comment comes from. Origin can be one of the following text strings: 'translator' 'developer', 'programmer', 'source code' (synonyms)

### classmethod buildfromunit (unit)

Build a native unit from a foreign unit, preserving as much information as possible.

### getcontext()

Get the message context.

### geterrors()

Get all error messages.

### **Return type** Dictionary

### getid()

A unique identifier for this unit.

### **Return type** string

**Returns** an identifier for this unit that is unique in the store

Derived classes should override this in a way that guarantees a unique identifier for each unit in the store.

### getlocations()

Returns the a list of the location(s) of the string.

### getnotes (origin=None)

Returns all notes about this unit.

It will probably be freeform text or something reasonable that can be synthesised by the format. It should not include location comments (see <code>getlocations()</code>).

### gettargetlen()

Returns the length of the target string.

### Return type Integer

**Note:** Plural forms might be combined.

### getunits()

This unit in a list.

### hasplural()

Tells whether or not this specific unit has plural strings.

#### infer state()

Empty method that should be overridden in sub-classes to infer the current state(\_n) of the unit from its current state.

### isblank()

Used to see if this unit has no source or target string.

**Note:** This is probably used more to find translatable units, and we might want to move in that direction rather and get rid of this.

### isfuzzy()

Indicates whether this unit is fuzzy.

#### isheader()

Indicates whether this unit is a header.

### isobsolete()

indicate whether a unit is obsolete

### isreview()

Indicates whether this unit needs review.

#### istranslatable()

Indicates whether this unit can be translated.

This should be used to distinguish real units for translation from header, obsolete, binary or other blank units.

### istranslated()

Indicates whether this unit is translated.

This should be used rather than deducing it from .target, to ensure that other classes can implement more functionality (as XLIFF does).

#### makeobsolete()

Make a unit obsolete

### markfuzzy(value=True)

Marks the unit as fuzzy or not.

### markreviewneeded (needsreview=True, explanation=None)

Marks the unit to indicate whether it needs review.

#### **Parameters**

- needsreview Defaults to True.
- **explanation** Adds an optional explanation as a note.

merge (otherunit, overwrite=False, comments=True, authoritative=False)

Do basic format agnostic merging.

### multistring\_to\_rich (mulstring)

Convert a multistring to a list of "rich" string trees:

```
>>> target = multistring([u'foo', u'bar', u'baz'])
>>> TranslationUnit.multistring_to_rich(target)
[<StringElem([<StringElem([u'foo'])>])>,
    <StringElem([<StringElem([u'bar'])>])>,
    <StringElem([<StringElem([u'baz'])>])>]
```

```
removenotes (origin=None)
          Remove all the translator's notes.
     rich_source
          See also:
          rich_to_multistring(), multistring_to_rich()
     rich_target
          See also:
          rich_to_multistring(), multistring_to_rich()
     classmethod rich_to_multistring(elem_list)
          Convert a "rich" string tree to a multistring:
         >>> from translate.storage.placeables.interfaces import X
         >>> rich = [StringElem(['foo', X(id='xxx', sub=[' ']), 'bar'])]
          >>> TranslationUnit.rich_to_multistring(rich)
          multistring(u'foo bar')
     setcontext (context)
          Set the message context
     setid(value)
         Sets the unique identified for this unit.
          only implemented if format allows ids independant from other unit properties like source or context
     unit_iter()
          Iterator that only returns this unit.
tmdb
Module to provide a translation memory database.
exception translate.storage.tmdb.LanguageError(value)
module for parsing TMX translation memeory files
class translate.storage.tmx.tmxfile(inputfile=None,
                                                                sourcelanguage='en',
                                                                                         target-
                                              language=None, **kwargs)
     Class representing a TMX file store.
     UnitClass
          alias of tmxunit
     add_unit_to_index(unit)
          Add a unit to source and location idexes
     addheader()
          Method to be overridden to initialise headers, etc.
     addsourceunit (source)
          Adds and returns a new unit with the given string as first entry.
```

tmx

```
addtranslation (source, srclang, translation, translang, comment=None)
     addtranslation method for testing old unit tests
addunit (unit, new=True)
     Append the given unit to the object's list of units.
     This method should always be used rather than trying to modify the list manually.
         Parameters unit (TranslationUnit) - The unit that will be added.
detect_encoding(text, default_encodings=None)
     Try to detect a file encoding from text, using either the chardet lib or by trying to decode the file.
fallback_detection (text)
     Simple detection based on BOM in case chardet is not available.
findid(id)
     find unit with matching id by checking id_index
findunit (source)
     Find the unit with the given source string.
         Return type TranslationUnit or None
findunits (source)
     Find the units with the given source string.
         Return type TranslationUnit or None
getids (filename=None)
     return a list of unit ids
getprojectstyle()
     Get the project type for this store.
getsourcelanguage()
     Get the source language for this store.
gettargetlanguage()
     Get the target language for this store.
getunits()
     Return a list of all units in this store.
initbody()
     Initialises self.body so it never needs to be retrieved from the XML again.
     Return True if the object doesn't contain any translation units.
makeindex()
     Indexes the items in this store. At least .sourceindex should be useful.
merge_on
     The matching criterion to use when merging on.
         Returns The default matching criterion for all the subclasses.
```

For example namespaced ("source") in an XLIFF document might return:

Return type string

Returns name in Clark notation.

namespaced (name)

```
{urn:oasis:names:tc:xliff:document:1.1}source
          This is needed throughout lxml.
     parse (xml)
          Populates this object from the given xml string
     classmethod parsefile (storefile)
          Reads the given file (or opens the given filename) and parses back to an object.
     classmethod parsestring(storestring)
          Convert the string representation back to an object.
     remove_unit_from_index(unit)
          Remove a unit from source and locaton indexes
     require_index()
          make sure source index exists
     save()
          Save to the file that data was originally read from, if available.
     savefile (storefile)
          Write the string representation to the given file (or filename).
     serialize(out=None)
          Converts to a string containing the file's XML
     setprojectstyle (project_style)
          Set the project type for this store.
     setsourcelanguage (sourcelanguage)
          Set the source language for this store.
     settargetlanguage (targetlanguage)
          Set the target language for this store.
     translate (sourcetext, sourcelang=None, targetlang=None)
          method to test old unit tests
     unit iter()
          Iterator over all the units in this store.
class translate.storage.tmx.tmxunit(source, empty=False, **kwargs)
     A single unit in the TMX file.
     adderror (errorname, errortext)
          Adds an error message to this unit.
     addlocation (location)
          Add one location to the list of locations.
```

**Note:** Shouldn't be implemented if the format doesn't support it.

#### addlocations (location)

Add a location or a list of locations.

**Note:** Most classes shouldn't need to implement this, but should rather implement TranslationUnit.addlocation().

Warning: This method might be removed in future.

### addnote (text, origin=None, position='append')

Add a note specifically in a "note" tag.

The origin parameter is ignored

### classmethod buildfromunit (unit)

Build a native unit from a foreign unit, preserving as much information as possible.

### copy()

Make a copy of the translation unit.

We don't want to make a deep copy - this could duplicate the whole XML tree. For now we just serialise and reparse the unit's XML.

### createlanguageNode (lang, text, purpose)

returns a langset xml Element setup with given parameters

### getNodeText (languageNode, xml\_space='preserve')

Retrieves the term from the given languageNode.

#### getcontext()

Get the message context.

### geterrors()

Get all error messages.

#### getid()

Returns the identifier for this unit. The optional tuid property is used if available, otherwise we inherit .getid(). Note that the tuid property is only mandated to be unique from TMX 2.0.

### getlanguageNode (lang=None, index=None)

Retrieves a languageNode either by language or by index.

### getlanguageNodes()

Returns a list of all nodes that contain per language information.

### getlocations()

A list of source code locations.

### Return type List

**Note:** Shouldn't be implemented if the format doesn't support it.

### getnotes (origin=None)

Returns all notes about this unit.

It will probably be freeform text or something reasonable that can be synthesised by the format. It should not include location comments (see <code>getlocations()</code>).

# gettarget (lang=None)

retrieves the "target" text (second entry), or the entry in the specified language, if it exists

## gettargetlen()

Returns the length of the target string.

### Return type Integer

**Note:** Plural forms might be combined.

### getunits()

This unit in a list.

### hasplural()

Tells whether or not this specific unit has plural strings.

### infer\_state()

Empty method that should be overridden in sub-classes to infer the current state(\_n) of the unit from its current state.

### isblank()

Used to see if this unit has no source or target string.

**Note:** This is probably used more to find translatable units, and we might want to move in that direction rather and get rid of this.

### isfuzzy()

Indicates whether this unit is fuzzy.

#### isheader()

Indicates whether this unit is a header.

#### isobsolete()

indicate whether a unit is obsolete

# isreview()

Indicates whether this unit needs review.

# istranslatable()

Indicates whether this unit can be translated.

This should be used to distinguish real units for translation from header, obsolete, binary or other blank units.

### istranslated()

Indicates whether this unit is translated.

This should be used rather than deducing it from .target, to ensure that other classes can implement more functionality (as XLIFF does).

### makeobsolete()

Make a unit obsolete

#### markfuzzy (value=True)

Marks the unit as fuzzy or not.

# $\verb|markreviewneeded| (needs review=True, explanation=None)$

Marks the unit to indicate whether it needs review.

### **Parameters**

- needsreview Defaults to True.
- **explanation** Adds an optional explanation as a note.

merge (otherunit, overwrite=False, comments=True, authoritative=False)

Do basic format agnostic merging.

### multistring\_to\_rich(mulstring)

Convert a multistring to a list of "rich" string trees:

### namespaced (name)

Returns name in Clark notation.

For example namespaced ("source") in an XLIFF document might return:

```
{urn:oasis:names:tc:xliff:document:1.1}source
```

This is needed throughout lxml.

```
removenotes (origin=None)
```

Remove all the translator notes.

```
rich_source
```

#### See also:

```
rich to multistring(), multistring to rich()
```

#### rich\_target

### See also:

```
rich_to_multistring(), multistring_to_rich()
```

# classmethod rich\_to\_multistring(elem\_list)

Convert a "rich" string tree to a multistring:

```
>>> from translate.storage.placeables.interfaces import X
>>> rich = [StringElem(['foo', X(id='xxx', sub=[' ']), 'bar'])]
>>> TranslationUnit.rich_to_multistring(rich)
multistring(u'foo bar')
```

#### setcontext (context)

Set the message context

### setid(value)

Sets the unique identified for this unit.

only implemented if format allows ids independant from other unit properties like source or context

```
settarget (target, lang='xx', append=False)
```

Sets the "target" string (second language), or alternatively appends to the list

### unit\_iter()

Iterator that only returns this unit.

# trados

Manage the Trados .txt Translation Memory format

A Trados file looks like this:

```
<TrU>
<CrD>18012000, 13:18:35
<CrU>CAROL-ANN
<UsC>0
<Seg L=EN_GB>Association for Road Safety \endash Conference
<Seg L=DE_DE>Tagung der Gesellschaft für Verkehrssicherheit
</TrU>
<TrU>
<CrD>18012000, 13:19:14
<Cru>CAROL-ANN
<UsC>0
<Seg L=EN_GB>Road Safety Education in our Schools
<Seg L=DE_DE>Verkehrserziehung an Schulen
translate.storage.trados.TRADOS_TIMEFORMAT = '%d%m%Y, %H:%M:%S'
     Time format used by Trados .txt
translate.storage.trados.RTF_ESCAPES = {u'\\-': u'\xad', u'\\_': u'\\2011', u'\\bullet':
     RTF control to Unicode map. See http://msdn.microsoft.com/en-us/library/aa140283(v=office.10).aspx
translate.storage.trados.escape(text)
     Convert Unicode string to Trodas escapes
translate.storage.trados.unescape(text)
     Convert Trados text to normal Unicode string
class translate.storage.trados.TradosTxtDate(newtime=None)
     Manages the timestamps in the Trados .txt format of DDMMYYY, hh:mm:ss
     get_time()
         Get the time_struct object
     get timestring()
         Get the time in the Trados time format
     set time(newtime)
         Set the time_struct object
             Parameters newtime (time.time_struct) - a new time object
     set_timestring(timestring)
         Set the time_struct object using a Trados time formated string
             Parameters timestring (String) - A Trados time string (DDMMYYYY, hh:mm:ss)
     time
         Get the time struct object
     timestring
         Get the time in the Trados time format
class translate.storage.trados.TradosUnit(source=None)
     adderror (errorname, errortext)
         Adds an error message to this unit.
             Parameters
                 • errorname (string) – A single word to id the error.
                 • errortext (string) – The text describing the error.
```

#### addlocation (location)

Add one location to the list of locations.

**Note:** Shouldn't be implemented if the format doesn't support it.

#### addlocations (location)

Add a location or a list of locations.

**Note:** Most classes shouldn't need to implement this, but should rather implement TranslationUnit.addlocation().

**Warning:** This method might be removed in future.

# addnote (text, origin=None, position='append')

Adds a note (comment).

#### **Parameters**

- **text** (*string*) Usually just a sentence or two.
- **origin** (string) Specifies who/where the comment comes from. Origin can be one of the following text strings: 'translator' 'developer', 'programmer', 'source code' (synonyms)

#### classmethod buildfromunit (unit)

Build a native unit from a foreign unit, preserving as much information as possible.

### getcontext()

Get the message context.

### geterrors()

Get all error messages.

Return type Dictionary

### getid()

A unique identifier for this unit.

### Return type string

Returns an identifier for this unit that is unique in the store

Derived classes should override this in a way that guarantees a unique identifier for each unit in the store.

### getlocations()

A list of source code locations.

### **Return type** List

**Note:** Shouldn't be implemented if the format doesn't support it.

### getnotes (origin=None)

Returns all notes about this unit.

It will probably be freeform text or something reasonable that can be synthesised by the format. It should not include location comments (see <code>getlocations()</code>).

### gettargetlen()

Returns the length of the target string.

### Return type Integer

Note: Plural forms might be combined.

#### getunits()

This unit in a list.

### hasplural()

Tells whether or not this specific unit has plural strings.

#### infer\_state()

Empty method that should be overridden in sub-classes to infer the current state(\_n) of the unit from its current state.

### isblank()

Used to see if this unit has no source or target string.

**Note:** This is probably used more to find translatable units, and we might want to move in that direction rather and get rid of this.

### isfuzzy()

Indicates whether this unit is fuzzy.

### isheader()

Indicates whether this unit is a header.

## isobsolete()

indicate whether a unit is obsolete

#### isreview()

Indicates whether this unit needs review.

### istranslatable()

Indicates whether this unit can be translated.

This should be used to distinguish real units for translation from header, obsolete, binary or other blank units.

### istranslated()

Indicates whether this unit is translated.

This should be used rather than deducing it from .target, to ensure that other classes can implement more functionality (as XLIFF does).

# makeobsolete()

Make a unit obsolete

# markfuzzy(value=True)

Marks the unit as fuzzy or not.

### markreviewneeded (needsreview=True, explanation=None)

Marks the unit to indicate whether it needs review.

# **Parameters**

• needsreview - Defaults to True.

• **explanation** – Adds an optional explanation as a note.

```
merge (otherunit, overwrite=False, comments=True, authoritative=False)
```

Do basic format agnostic merging.

```
multistring_to_rich(mulstring)
```

Convert a multistring to a list of "rich" string trees:

```
>>> target = multistring([u'foo', u'bar', u'baz'])
>>> TranslationUnit.multistring_to_rich(target)
[<StringElem([<StringElem([u'foo'])>])>,
    <StringElem([<StringElem([u'bar'])>])>,
    <StringElem([<StringElem([u'baz'])>])>]
```

# removenotes (origin=None)

Remove all the translator's notes.

rich\_source

# See also:

```
rich_to_multistring(), multistring_to_rich()
```

rich\_target

### See also:

```
rich_to_multistring(), multistring_to_rich()
```

## classmethod rich\_to\_multistring(elem\_list)

Convert a "rich" string tree to a multistring:

```
>>> from translate.storage.placeables.interfaces import X
>>> rich = [StringElem(['foo', X(id='xxx', sub=[' ']), 'bar'])]
>>> TranslationUnit.rich_to_multistring(rich)
multistring(u'foo bar')
```

# setcontext (context)

Set the message context

#### setid(value)

Sets the unique identified for this unit.

only implemented if format allows ids independant from other unit properties like source or context

# unit\_iter()

Iterator that only returns this unit.

```
class translate.storage.trados.TradosTxtTmFile(inputfile=None, **kwargs)
```

A Trados translation memory file

# UnitClass

alias of TradosUnit

# add\_unit\_to\_index(unit)

Add a unit to source and location idexes

#### addsourceunit (source)

Add and returns a new unit with the given source string.

Return type TranslationUnit

#### addunit (unit)

Append the given unit to the object's list of units.

This method should always be used rather than trying to modify the list manually.

**Parameters unit** (TranslationUnit) – The unit that will be added.

#### detect\_encoding(text, default\_encodings=None)

Try to detect a file encoding from *text*, using either the chardet lib or by trying to decode the file.

### fallback\_detection (text)

Simple detection based on BOM in case chardet is not available.

### findid(id)

find unit with matching id by checking id\_index

#### findunit (source)

Find the unit with the given source string.

Return type TranslationUnit or None

#### findunits (source)

Find the units with the given source string.

Return type TranslationUnit or None

### getids (filename=None)

return a list of unit ids

### getprojectstyle()

Get the project type for this store.

### getsourcelanguage()

Get the source language for this store.

# gettargetlanguage()

Get the target language for this store.

# getunits()

Return a list of all units in this store.

### isempty()

Return True if the object doesn't contain any translation units.

## makeindex()

Indexes the items in this store. At least .sourceindex should be useful.

# merge\_on

The matching criterion to use when merging on.

**Returns** The default matching criterion for all the subclasses.

Return type string

# parse (input)

parser to process the given source string

### classmethod parsefile (storefile)

Reads the given file (or opens the given filename) and parses back to an object.

### classmethod parsestring(storestring)

Convert the string representation back to an object.

# ${\tt remove\_unit\_from\_index}\,(\mathit{unit})$

Remove a unit from source and locaton indexes

```
require index()
           make sure source index exists
     save()
           Save to the file that data was originally read from, if available.
     savefile (storefile)
           Write the string representation to the given file (or filename).
           Converts to a bytes representation that can be parsed back using parsestring(). out should be an
           open file-like objects to write to.
     setprojectstyle (project_style)
           Set the project type for this store.
     setsourcelanguage (sourcelanguage)
           Set the source language for this store.
     settargetlanguage (targetlanguage)
           Set the target language for this store.
     translate(source)
           Return the translated string for a given source string.
               Return type String or None
     unit iter()
           Iterator over all the units in this store.
Module for handling Qt linguist (.ts) files.
This will eventually replace the older ts.py which only supports the older format. While converters haven't been
updated to use this module, we retain both.
TS file format 4.3, 4.8, 5. Example.
Specification of the valid variable entries, 2
class translate.storage.ts2.tsfile(*args, **kwargs)
     Class representing a TS file store.
     UnitClass
           alias of tsunit
     add unit to index (unit)
           Add a unit to source and location idexes
     addheader()
           Method to be overridden to initialise headers, etc.
     addsourceunit (source)
           Adds and returns a new unit with the given string as first entry.
     addunit (unit, new=True, contextname=None, createifmissing=True)
           Adds the given unit to the last used body node (current context).
           If the contextname is specified, switch to that context (creating it if allowed by createifmissing).
     detect_encoding(text, default_encodings=None)
```

ts2

4.1. API 645

Try to detect a file encoding from text, using either the chardet lib or by trying to decode the file.

#### fallback detection (text)

Simple detection based on BOM in case chardet is not available.

#### findid(id)

find unit with matching id by checking id\_index

### findunit (source)

Find the unit with the given source string.

Return type TranslationUnit or None

# findunits (source)

Find the units with the given source string.

Return type TranslationUnit or None

#### getids (filename=None)

return a list of unit ids

# getprojectstyle()

Get the project type for this store.

#### getsourcelanguage()

Get the source language for this .ts file.

The 'sourcelanguage' attribute was only added to the TS format in Qt v4.5. We return 'en' if there is no sourcelanguage set.

We don't implement setsourcelanguage as users really shouldn't be altering the source language in .ts files, it should be set correctly by the extraction tools.

Returns ISO code e.g. af, fr, pt\_BR

Return type String

# gettargetlanguage()

Get the target language for this .ts file.

Returns ISO code e.g. af, fr, pt\_BR

Return type String

#### getunits()

Return a list of all units in this store.

# initbody()

Initialises self.body.

### isempty()

Return True if the object doesn't contain any translation units.

#### makeindex()

Indexes the items in this store. At least .sourceindex should be useful.

# merge\_on

The matching criterion to use when merging on.

**Returns** The default matching criterion for all the subclasses.

Return type string

# namespaced(name)

Returns name in Clark notation.

For example namespaced ("source") in an XLIFF document might return:

```
{urn:oasis:names:tc:xliff:document:1.1}source
          This is needed throughout lxml.
     parse (xml)
          Populates this object from the given xml string
     classmethod parsefile (storefile)
          Reads the given file (or opens the given filename) and parses back to an object.
     classmethod parsestring(storestring)
          Convert the string representation back to an object.
     remove_unit_from_index(unit)
          Remove a unit from source and locaton indexes
     require_index()
          make sure source index exists
     save()
          Save to the file that data was originally read from, if available.
     savefile (storefile)
          Write the string representation to the given file (or filename).
     serialize(out)
          Write the XML document to a file.
     setprojectstyle (project_style)
          Set the project type for this store.
     setsourcelanguage (sourcelanguage)
          Set the source language for this store.
     settargetlanguage (targetlanguage)
          Set the target language for this .ts file to targetlanguage.
               Parameters targetlanguage (String) – ISO code e.g. af, fr, pt_BR
     translate(source)
          Return the translated string for a given source string.
               Return type String or None
     unit iter()
          Iterator over all the units in this store.
class translate.storage.ts2.tsunit(source, empty=False, **kwargs)
     A single term in the TS file.
     adderror (errorname, errortext)
          Adds an error message to this unit.
               Parameters
                   • errorname (string) – A single word to id the error.
                   • errortext (string) – The text describing the error.
     addlocation (location)
          Add one location to the list of locations.
```

Note: Shouldn't be implemented if the format doesn't support it.

### addlocations (location)

Add a location or a list of locations.

**Note:** Most classes shouldn't need to implement this, but should rather implement TranslationUnit.addlocation().

Warning: This method might be removed in future.

### addnote (text, origin=None, position='append')

Add a note specifically in the appropriate comment tag

### classmethod buildfromunit (unit)

Build a native unit from a foreign unit, preserving as much information as possible.

#### createlanguageNode (lang, text, purpose)

Returns an xml Element setup with given parameters.

### getNodeText (languageNode, xml\_space='preserve')

Retrieves the term from the given languageNode.

#### getcontext()

Get the message context.

### geterrors()

Get all error messages.

#### **Return type** Dictionary

### getid()

A unique identifier for this unit.

### Return type string

**Returns** an identifier for this unit that is unique in the store

Derived classes should override this in a way that guarantees a unique identifier for each unit in the store.

### getlanguageNode (lang=None, index=None)

Retrieves a language Node either by language or by index.

# getlanguageNodes()

We override this to get source and target nodes.

#### getlocations()

A list of source code locations.

# Return type List

**Note:** Shouldn't be implemented if the format doesn't support it.

#### getnotes (origin=None)

Returns all notes about this unit.

It will probably be freeform text or something reasonable that can be synthesised by the format. It should not include location comments (see <code>getlocations()</code>).

### gettarget (\*\*kwargs)

retrieves the "target" text (second entry), or the entry in the specified language, if it exists

#### gettargetlen()

Returns the length of the target string.

#### **Return type** Integer

**Note:** Plural forms might be combined.

### getunits()

This unit in a list.

#### hasplural()

Tells whether or not this specific unit has plural strings.

#### infer state()

Empty method that should be overridden in sub-classes to infer the current state(\_n) of the unit from its current state.

#### isblank()

Used to see if this unit has no source or target string.

**Note:** This is probably used more to find translatable units, and we might want to move in that direction rather and get rid of this.

# isfuzzy()

Indicates whether this unit is fuzzy.

#### isheader()

Indicates whether this unit is a header.

#### isobsolete()

indicate whether a unit is obsolete

# isreview()

States whether this unit needs to be reviewed

#### istranslatable()

Indicates whether this unit can be translated.

This should be used to distinguish real units for translation from header, obsolete, binary or other blank units.

# istranslated()

Indicates whether this unit is translated.

This should be used rather than deducing it from .target, to ensure that other classes can implement more functionality (as XLIFF does).

### makeobsolete()

Make a unit obsolete

### markfuzzy(value=True)

Marks the unit as fuzzy or not.

#### markreviewneeded (needsreview=True, explanation=None)

Marks the unit to indicate whether it needs review.

#### **Parameters**

- needsreview Defaults to True.
- **explanation** Adds an optional explanation as a note.

merge (otherunit, overwrite=False, comments=True, authoritative=False)

Do basic format agnostic merging.

#### multistring\_to\_rich(mulstring)

Convert a multistring to a list of "rich" string trees:

#### namespaced(name)

Returns name in Clark notation.

For example namespaced ("source") in an XLIFF document might return:

```
{urn:oasis:names:tc:xliff:document:1.1}source
```

This is needed throughout lxml.

# removenotes (origin=None)

Remove all the translator notes.

rich source

# See also:

```
rich_to_multistring(), multistring_to_rich()
```

# rich\_target

### See also:

```
rich_to_multistring(), multistring_to_rich()
```

# classmethod rich\_to\_multistring(elem\_list)

Convert a "rich" string tree to a multistring:

```
>>> from translate.storage.placeables.interfaces import X
>>> rich = [StringElem(['foo', X(id='xxx', sub=[' ']), 'bar'])]
>>> TranslationUnit.rich_to_multistring(rich)
multistring(u'foo bar')
```

### setcontext(value)

Set the message context

#### setid(value)

Sets the unique identified for this unit.

only implemented if format allows ids independant from other unit properties like source or context

```
settarget (target, lang='xx', append=False)
           Sets the "target" string (second language), or alternatively appends to the list
     statemap = {None: 100, '': 100, 'obsolete': -100, 'unfinished':
                                                                                                    30, 'vanished':
           This maps the unit "type" attribute to state.
     unit iter()
           Iterator that only returns this unit.
Module for parsing Qt .ts files for translation.
Currently this module supports the old format of .ts files. Some applications use the newer .ts format which are
documented here: TS file format 4.3, Example
Specification of the valid variable entries, 2
This class implements the functionality for handling plain text files, or similar wiki type files.
Supported formats are
        · Plain text
        · dokuwiki
        · MediaWiki
class translate.storage.txt.TxtFile(inputfile=None, flavour=None, no_segmentation=False,
                                                  **kwargs)
     This class represents a text file, made up of txtunits
     UnitClass
           alias of TxtUnit
     add_unit_to_index(unit)
           Add a unit to source and location idexes
     addsourceunit (source)
           Add and returns a new unit with the given source string.
               Return type TranslationUnit
     addunit (unit)
           Append the given unit to the object's list of units.
           This method should always be used rather than trying to modify the list manually.
               Parameters unit (TranslationUnit) – The unit that will be added.
     detect_encoding(text, default_encodings=None)
           Try to detect a file encoding from text, using either the chardet lib or by trying to decode the file.
```

ts

txt

fallback\_detection (text)

findid(id)

4.1. API 651

Simple detection based on BOM in case chardet is not available.

find unit with matching id by checking id\_index

```
findunit (source)
     Find the unit with the given source string.
         Return type TranslationUnit or None
findunits (source)
     Find the units with the given source string.
         Return type TranslationUnit or None
getids (filename=None)
     return a list of unit ids
getprojectstyle()
     Get the project type for this store.
getsourcelanguage()
     Get the source language for this store.
gettargetlanguage()
     Get the target language for this store.
getunits()
     Return a list of all units in this store.
isempty()
     Return True if the object doesn't contain any translation units.
makeindex()
     Indexes the items in this store. At least .sourceindex should be useful.
merge_on
     The matching criterion to use when merging on.
         Returns The default matching criterion for all the subclasses.
         Return type string
parse (lines)
     Read in text lines and create txtunits from the blocks of text
classmethod parsefile(storefile)
     Reads the given file (or opens the given filename) and parses back to an object.
classmethod parsestring(storestring)
     Convert the string representation back to an object.
remove_unit_from_index(unit)
     Remove a unit from source and locaton indexes
require_index()
     make sure source index exists
save()
     Save to the file that data was originally read from, if available.
savefile (storefile)
     Write the string representation to the given file (or filename).
serialize (out)
     Converts to a bytes representation that can be parsed back using parsestring (). out should be an
```

open file-like objects to write to.

setprojectstyle (*project\_style*)
Set the project type for this store.

### setsourcelanguage (sourcelanguage)

Set the source language for this store.

# settargetlanguage (targetlanguage)

Set the target language for this store.

#### translate(source)

Return the translated string for a given source string.

Return type String or None

### unit\_iter()

Iterator over all the units in this store.

#### class translate.storage.txt.TxtUnit (source=", \*\*kwargs)

This class represents a block of text from a text file

# adderror (errorname, errortext)

Adds an error message to this unit.

#### **Parameters**

- **errorname** (*string*) A single word to id the error.
- **errortext** (*string*) The text describing the error.

#### addlocation (location)

Add one location to the list of locations.

**Note:** Shouldn't be implemented if the format doesn't support it.

# addlocations (location)

Add a location or a list of locations.

**Note:** Most classes shouldn't need to implement this, but should rather implement TranslationUnit.addlocation().

**Warning:** This method might be removed in future.

addnote (text, origin=None, position='append')

Adds a note (comment).

#### **Parameters**

- **text** (*string*) Usually just a sentence or two.
- **origin** (*string*) Specifies who/where the comment comes from. Origin can be one of the following text strings: 'translator' 'developer', 'programmer', 'source code' (synonyms)

# ${\tt classmethod\ buildfromunit}\ (\mathit{unit})$

Build a native unit from a foreign unit, preserving as much information as possible.

### getcontext()

Get the message context.

#### geterrors()

Get all error messages.

### **Return type** Dictionary

#### getid()

A unique identifier for this unit.

### Return type string

**Returns** an identifier for this unit that is unique in the store

Derived classes should override this in a way that guarantees a unique identifier for each unit in the store.

### getlocations()

A list of source code locations.

#### **Return type** List

**Note:** Shouldn't be implemented if the format doesn't support it.

### getnotes (origin=None)

Returns all notes about this unit.

It will probably be freeform text or something reasonable that can be synthesised by the format. It should not include location comments (see getlocations()).

# gettargetlen()

Returns the length of the target string.

#### **Return type** Integer

**Note:** Plural forms might be combined.

# getunits()

This unit in a list.

### hasplural()

Tells whether or not this specific unit has plural strings.

#### infer\_state()

Empty method that should be overridden in sub-classes to infer the current state(\_n) of the unit from its current state.

# isblank()

Used to see if this unit has no source or target string.

**Note:** This is probably used more to find translatable units, and we might want to move in that direction rather and get rid of this.

### isfuzzy()

Indicates whether this unit is fuzzy.

#### isheader()

Indicates whether this unit is a header.

### isobsolete()

indicate whether a unit is obsolete

### isreview()

Indicates whether this unit needs review.

#### istranslatable()

Indicates whether this unit can be translated.

This should be used to distinguish real units for translation from header, obsolete, binary or other blank units.

# istranslated()

Indicates whether this unit is translated.

This should be used rather than deducing it from .target, to ensure that other classes can implement more functionality (as XLIFF does).

### makeobsolete()

Make a unit obsolete

### markfuzzy(value=True)

Marks the unit as fuzzy or not.

#### markreviewneeded (needsreview=True, explanation=None)

Marks the unit to indicate whether it needs review.

#### **Parameters**

- needsreview Defaults to True.
- **explanation** Adds an optional explanation as a note.

merge (otherunit, overwrite=False, comments=True, authoritative=False)

Do basic format agnostic merging.

#### multistring to rich(mulstring)

Convert a multistring to a list of "rich" string trees:

# removenotes (origin=None)

Remove all the translator's notes.

#### rich source

### See also:

```
rich_to_multistring(), multistring_to_rich()
```

# rich\_target

#### See also:

```
rich_to_multistring(), multistring_to_rich()
```

### classmethod rich\_to\_multistring(elem\_list)

Convert a "rich" string tree to a multistring:

```
>>> from translate.storage.placeables.interfaces import X
>>> rich = [StringElem(['foo', X(id='xxx', sub=[' ']), 'bar'])]
>>> TranslationUnit.rich_to_multistring(rich)
multistring(u'foo bar')
```

```
setcontext (context)
    Set the message context

setid(value)
    Sets the unique identified for this unit.
    only implemented if format allows ids independant from other unit properties like source or context

target
    gets the unquoted target string

unit_iter()
    Iterator that only returns this unit.
```

#### utx

Manage the Universal Terminology eXchange (UTX) format

UTX is a format for terminology exchange, designed it seems with Machine Translation (MT) as it's primary consumer. The format is created by the Asia-Pacific Association for Machine Translation (AAMT).

It is a bilingual base class derived format with UtxFile and UtxUnit providing file and unit level access.

The format can manage monolingual dictionaries but these classes don't implement that.

**Specification** The format is implemented according to UTX v1.0 (No longer available from their website. The current UTX version may be downloaded instead).

**Format Implementation** The UTX format is a Tab Separated Value (TSV) file in UTF-8. The first two lines are headers with subsequent lines containing a single source target definition.

**Encoding** The files are UTF-8 encoded with no BOM and CR+LF line terminators.

```
class translate.storage.utx.UtxDialect
   Describe the properties of an UTX generated TAB-delimited dictionary file.
   classmethod combine (dialect, fmtparams)
        Create a new dialect with defaults and added parameters.
```

class translate.storage.utx.UtxFile(inputfile=None, \*\*kwargs)
 A UTX dictionary file

```
UnitClass
```

alias of UtxUnit

### add\_unit\_to\_index(unit)

Add a unit to source and location idexes

#### addsourceunit (source)

Add and returns a new unit with the given source string.

Return type TranslationUnit

### addunit (unit)

Append the given unit to the object's list of units.

This method should always be used rather than trying to modify the list manually.

Parameters unit (TranslationUnit) - The unit that will be added.

```
detect_encoding (text, default_encodings=None)
```

Try to detect a file encoding from text, using either the chardet lib or by trying to decode the file.

```
fallback detection(text)
     Simple detection based on BOM in case chardet is not available.
findid(id)
     find unit with matching id by checking id_index
findunit (source)
     Find the unit with the given source string.
         Return type TranslationUnit or None
findunits (source)
     Find the units with the given source string.
         Return type TranslationUnit or None
getids (filename=None)
     return a list of unit ids
getprojectstyle()
     Get the project type for this store.
getsourcelanguage()
     Get the source language for this store.
gettargetlanguage()
     Get the target language for this store.
getunits()
     Return a list of all units in this store.
isempty()
     Return True if the object doesn't contain any translation units.
makeindex()
     Indexes the items in this store. At least .sourceindex should be useful.
merge_on
     The matching criterion to use when merging on.
         Returns The default matching criterion for all the subclasses.
         Return type string
parse(input)
     parsese the given file or file source string
classmethod parsefile(storefile)
     Reads the given file (or opens the given filename) and parses back to an object.
classmethod parsestring(storestring)
     Convert the string representation back to an object.
remove_unit_from_index(unit)
     Remove a unit from source and locaton indexes
require_index()
     make sure source index exists
save()
     Save to the file that data was originally read from, if available.
savefile (storefile)
     Write the string representation to the given file (or filename).
```

#### serialize(out)

Converts to a bytes representation that can be parsed back using parsestring(). out should be an open file-like objects to write to.

### setprojectstyle (project\_style)

Set the project type for this store.

#### setsourcelanguage (sourcelanguage)

Set the source language for this store.

# settargetlanguage (targetlanguage)

Set the target language for this store.

#### translate(source)

Return the translated string for a given source string.

### Return type String or None

### unit\_iter()

Iterator over all the units in this store.

#### class translate.storage.utx.UtxHeader

A UTX header entry

**A UTX header is a single line that looks like this::** #UTX-S <version>; < source language >/< target language>; <date created>; <optional fields (creator, license, etc.)>

### Where::

- UTX-S version is currently 1.00.
- Source language/target language: ISO 639, 3166 formats. In the case of monolingual dictionary, target language should be omitted.
- Date created: ISO 8601 format
- Optional fields (creator, license, etc.)

```
class translate.storage.utx.UtxUnit(source=None)
```

A UTX dictionary unit

#### adderror (errorname, errortext)

Adds an error message to this unit.

#### **Parameters**

- **errorname** (*string*) A single word to id the error.
- **errortext** (*string*) The text describing the error.

### addlocation (location)

Add one location to the list of locations.

**Note:** Shouldn't be implemented if the format doesn't support it.

#### addlocations (location)

Add a location or a list of locations.

Note: Most classes shouldn't need to implement this, but should rather implement TranslationUnit.addlocation().

Warning: This method might be removed in future.

# addnote (text, origin=None, position=u'append')

Adds a note (comment).

#### **Parameters**

- **text** (*string*) Usually just a sentence or two.
- **origin** (*string*) Specifies who/where the comment comes from. Origin can be one of the following text strings: 'translator' 'developer', 'programmer', 'source code' (synonyms)

### classmethod buildfromunit (unit)

Build a native unit from a foreign unit, preserving as much information as possible.

#### dict

Get the dictionary of values for a UTX line

#### getcontext()

Get the message context.

### getdict()

Get the dictionary of values for a UTX line

# geterrors()

Get all error messages.

### **Return type** Dictionary

### getid()

A unique identifier for this unit.

#### **Return type** string

**Returns** an identifier for this unit that is unique in the store

Derived classes should override this in a way that guarantees a unique identifier for each unit in the store.

### getlocations()

A list of source code locations.

### Return type List

Note: Shouldn't be implemented if the format doesn't support it.

### getnotes (origin=None)

Returns all notes about this unit.

It will probably be freeform text or something reasonable that can be synthesised by the format. It should not include location comments (see <code>qetlocations()</code>).

# gettargetlen()

Returns the length of the target string.

### Return type Integer

Note: Plural forms might be combined.

#### getunits()

This unit in a list.

### hasplural()

Tells whether or not this specific unit has plural strings.

#### infer\_state()

Empty method that should be overridden in sub-classes to infer the current state(\_n) of the unit from its current state.

#### isblank()

Used to see if this unit has no source or target string.

**Note:** This is probably used more to find translatable units, and we might want to move in that direction rather and get rid of this.

### isfuzzy()

Indicates whether this unit is fuzzy.

#### isheader()

Indicates whether this unit is a header.

#### isobsolete()

indicate whether a unit is obsolete

#### isreview()

Indicates whether this unit needs review.

### istranslatable()

Indicates whether this unit can be translated.

This should be used to distinguish real units for translation from header, obsolete, binary or other blank units.

#### istranslated()

Indicates whether this unit is translated.

This should be used rather than deducing it from .target, to ensure that other classes can implement more functionality (as XLIFF does).

### makeobsolete()

Make a unit obsolete

# markfuzzy(value=True)

Marks the unit as fuzzy or not.

### markreviewneeded (needsreview=True, explanation=None)

Marks the unit to indicate whether it needs review.

#### **Parameters**

- needsreview Defaults to True.
- **explanation** Adds an optional explanation as a note.

**merge** (otherunit, overwrite=False, comments=True, authoritative=False)

Do basic format agnostic merging.

# multistring\_to\_rich(mulstring)

Convert a multistring to a list of "rich" string trees:

### removenotes (origin=None)

Remove all the translator's notes.

#### rich source

#### See also:

```
rich_to_multistring(), multistring_to_rich()
```

### rich\_target

#### See also:

```
rich_to_multistring(), multistring_to_rich()
```

### classmethod rich\_to\_multistring(elem\_list)

Convert a "rich" string tree to a multistring:

```
>>> from translate.storage.placeables.interfaces import X
>>> rich = [StringElem(['foo', X(id='xxx', sub=[' ']), 'bar'])]
>>> TranslationUnit.rich_to_multistring(rich)
multistring(u'foo bar')
```

# setcontext (context)

Set the message context

#### setdict (newdict)

Set the dictionary of values for a UTX line

Parameters newdict (Dict) – a new dictionary with UTX line elements

#### setid(value)

Sets the unique identified for this unit.

only implemented if format allows ids independant from other unit properties like source or context

### unit\_iter()

Iterator that only returns this unit.

# versioncontrol

This module manages interaction with version control systems.

To implement support for a new version control system, inherit from GenericRevisionControlSystem.

### TODO:

- · Add authentication handling
- commitdirectory () should do a single commit instead of one for each file
- Maybe implement some caching for get\_versioned\_object() check profiler

```
translate.storage.versioncontrol.DEFAULT_RCS = ['svn', 'cvs', 'darcs', 'git', 'bzr', 'hq']
     the names of all supported revision control systems
     modules of the same name containing a class with the same name are expected to be defined below 'trans-
     late.storage.versioncontrol'
class translate.storage.versioncontrol.GenericRevisionControlSystem(location,
                                                                                          est parent=None)
     Bases: object
     The super class for all version control classes.
     Always inherit from this class to implement another RC interface.
     At least the two attributes RCS_METADIR and SCAN_PARENTS must be overriden by all implementations that
     derive from this class.
     By default, all implementations can rely on the following attributes:
            • root_dir: the parent of the metadata directory of the working copy
            • location_abs: the absolute path of the RCS object
            • location_rel: the path of the RCS object relative to root_dir
     RCS_METADIR = None
          The name of the metadata directory of the RCS
          e.g.: for Subversion -> ".svn"
     SCAN_PARENTS = None
          Whether to check the parent directories for the metadata directory of the RCS working copy
          Some revision control systems store their metadata directory only in the base of the working copy (e.g.
          bzr, GIT and Darcs) use True for these RCS
          Other RCS store a metadata directory in every single directory of the working copy (e.g. Subversion and
          CVS) use False for these RCS
     add (files, message=None, author=None)
          Dummy to be overridden by real implementations
     commit (message=None, author=None)
          Dummy to be overridden by real implementations
     getcleanfile (revision=None)
          Dummy to be overridden by real implementations
     update (revision=None, needs_revert=True)
          Dummy to be overridden by real implementations
translate.storage.versioncontrol.commitdirectory(directory,
                                                                             message=None,
                                                                                               au-
                                                                 thor=None)
     Commit all files below the given directory.
     Files that are just symlinked into the directory are supported, too
translate.storage.versioncontrol.get available version control systems()
     return the class objects of all locally available version control systems
translate.storage.versioncontrol.get_versioned_object(location,
                                                                                           version-
                                                                       ing systems=None,
                                                                                               fol-
                                                                       low symlinks=True,
                                                                                               old-
```

*est\_parent=None*)

return a versioned object for the given file

```
translate.storage.versioncontrol.get_versioned_objects_recursive(location,
                                                                                      version-
                                                                                      ing systems=None,
                                                                                      fol-
                                                                                      low_symlinks=True)
     return a list of objects, each pointing to a file below this directory
translate.storage.versioncontrol.run_command(command,cwd=None)
     Runs a command (array of program name and arguments) and returns the exitcode, the output and the error as a
     tuple.
          Parameters
                • command (list) – list of arguments to be joined for a program call
                • cwd(str) – optional directory where the command should be executed
translate.storage.versioncontrol.updatedirectory(directory)
     Update all files below the given directory.
     Files that are just symlinked into the directory are supported, too
bzr
class translate.storage.versioncontrol.bzr.bzr(location, oldest_parent=None)
     Class to manage items under revision control of bzr.
     add (files, message=None, author=None)
          Add and commit files.
     commit (message=None, author=None)
          Commits the file and supplies the given commit message if present
     getcleanfile (revision=None)
          Get a clean version of a file from the bzr repository
     update (revision=None, needs_revert=True)
          Does a clean update of the given path
translate.storage.versioncontrol.bzr.get_version()
     return a tuple of (major, minor) for the installed bazaar client
translate.storage.versioncontrol.bzr.is available()
     check if bzr is installed
CVS
class translate.storage.versioncontrol.cvs.cvs(location, oldest parent=None)
     Class to manage items under revision control of CVS.
     add (files, message=None, author=None)
          Add and commit the new files.
     commit (message=None, author=None)
          Commits the file and supplies the given commit message if present
          the 'author' parameter is not suitable for CVS, thus it is ignored
     getcleanfile (revision=None)
          Get the content of the file for the given revision
```

```
update (revision=None, needs_revert=True)
          Does a clean update of the given path
translate.storage.versioncontrol.cvs.is_available()
     check if cvs is installed
darcs
class translate.storage.versioncontrol.darcs.darcs(location, oldest_parent=None)
     Class to manage items under revision control of darcs.
     add (files, message=None, author=None)
          Add and commit files.
     commit (message=None, author=None)
          Commits the file and supplies the given commit message if present
     getcleanfile (revision=None)
          Get a clean version of a file from the darcs repository
              Parameters revision – ignored for darcs
     update (revision=None, needs_revert=True)
          Does a clean update of the given path
              Parameters revision – ignored for darcs
translate.storage.versioncontrol.darcs.is_available()
     check if darcs is installed
git
class translate.storage.versioncontrol.git.git(location, oldest_parent=None)
     Class to manage items under revision control of git.
     add (files, message=None, author=None)
          Add and commit the new files.
     commit (message=None, author=None, add=True)
          Commits the file and supplies the given commit message if present
     getcleanfile (revision=None)
          Get a clean version of a file from the git repository
     update (revision=None, needs revert=True)
          Does a clean update of the given path
\verb|translate.storage.versioncontrol.git.is_available()|\\
     check if git is installed
hg
translate.storage.versioncontrol.hg.get_version()
     Return a tuple of (major, minor) for the installed mercurial client.
class translate.storage.versioncontrol.hg.hg(location, oldest_parent=None)
     Class to manage items under revision control of mercurial.
```

```
add (files, message=None, author=None)
          Add and commit the new files.
     commit (message=None, author=None)
          Commits the file and supplies the given commit message if present
     getcleanfile (revision=None)
          Get a clean version of a file from the hg repository
     update (revision=None, needs revert=True)
          Does a clean update of the given path
              Parameters revision - ignored for hg
translate.storage.versioncontrol.hg.is_available()
     check if hg is installed
svn
translate.storage.versioncontrol.svn.get version()
     return a tuple of (major, minor) for the installed subversion client
translate.storage.versioncontrol.svn.is_available()
     check if svn is installed
class translate.storage.versioncontrol.svn.svn(location, oldest parent=None)
     Class to manage items under revision control of Subversion.
     add (files, message=None, author=None)
          Add and commit the new files.
     commit (message=None, author=None)
          commit the file and return the given message if present
          the 'author' parameter is used for revision property 'translate:author'
     getcleanfile (revision=None)
          return the content of the 'head' revision of the file
     update (revision=None, needs revert=True)
          update the working copy - remove local modifications if necessary
```

# wordfast

Manage the Wordfast Translation Memory format

Wordfast TM format is the Translation Memory format used by the Wordfast computer aided translation tool.

It is a bilingual base class derived format with <code>WordfastTMFile</code> and <code>WordfastUnit</code> providing file and unit level access.

Wordfast is a computer aided translation tool. It is an application built on top of Microsoft Word and is implemented as a rather sophisticated set of macros. Understanding that helps us understand many of the seemingly strange choices around this format including: encoding, escaping and file naming.

**Implementation** The implementation covers the full requirements of a Wordfast TM file. The files are simple Tab Separated Value (TSV) files that can be read by Microsoft Excel and other spreadsheet programs. They use the .txt extension which does make it more difficult to automatically identify such files.

The dialect of the TSV files is specified by WordfastDialect.

**Encoding** The files are UTF-16 or ISO-8859-1 (Latin1) encoded. These choices are most likely because Microsoft Word is the base editing tool for Wordfast.

The format is tab separated so we are able to detect UTF-16 vs Latin-1 by searching for the occurance of a UTF-16 tab character and then continuing with the parsing.

- **Timestamps** *WordfastTime* allows for the correct management of the Wordfast YYYYMMDD~HHMMSS timestamps. However, timestamps on individual units are not updated when edited.
- **Header** *WordfastHeader* provides header management support. The header functionality is fully implemented through observing the behaviour of the files in real use cases, input from the Wordfast programmers and public documentation.

**Escaping** Wordfast TM implements a form of escaping that covers two aspects:

- 1. Placeable: bold, formating, etc. These are left as is and ignored. It is up to the editor and future placeable implementation to manage these.
- 2. Escapes: items that may confuse Excel or translators are escaped as & 'XX;. These are fully implemented and are converted to and from Unicode. By observing behaviour and reading documentation we where able to observe all possible escapes. Unfortunately the escaping differs slightly between Windows and Mac version. This might cause errors in future. Functions allow for <\_wf\_to\_char> and back to Wordfast escape (< char to wf>).

**Extended Attributes** The last 4 columns allow users to define and manage extended attributes. These are left as is and are not directly managed byour implementation.

```
translate.storage.wordfast.TAB UTF16 = '\x00\t'
    The tab t character as it would appear in UTF-16 encoding
translate.storage.wordfast.WF_ESCAPE_MAP = ((u"&'26;", u'&'), (u"&'82;", u'\u201a'), (u"&'
    Mapping of Wordfast &'XX; escapes to correct Unicode characters
translate.storage.wordfast.WF_FIELDNAMES = [u'date', u'user', u'reuse', u'src-lang', u'sou
    Field names for a Wordfast TU
translate.storage.wordfast.WF_FIELDNAMES_HEADER = [u'date', u'userlist', u'tucount', u'src
    Field names for the Wordfast header
translate.storage.wordfast.WF_FIELDNAMES_HEADER_DEFAULTS = {u'attr1list': u'', u'attr2list
    Default or minimum header entries for a Wordfast file
translate.storage.wordfast.WF TIMEFORMAT = u'%Y%m%d~%H%M%S'
    Time format used by Wordfast
class translate.storage.wordfast.WordfastDialect
    Describe the properties of a Wordfast generated TAB-delimited file.
    classmethod combine (dialect, fmtparams)
         Create a new dialect with defaults and added parameters.
```

```
class translate.storage.wordfast.WordfastHeader(header=None)
```

A wordfast translation memory header

```
getheader()
```

Get the header dictionary

### header

Get the header dictionary

```
class translate.storage.wordfast.WordfastTMFile(inputfile=None, **kwargs)
    A Wordfast translation memory file
```

# UnitClass alias of WordfastUnit add\_unit\_to\_index(unit) Add a unit to source and location idexes addsourceunit (source) Add and returns a new unit with the given source string. Return type TranslationUnit addunit (unit) Append the given unit to the object's list of units. This method should always be used rather than trying to modify the list manually. **Parameters unit** (TranslationUnit) – The unit that will be added. detect\_encoding (text, default\_encodings=None) Try to detect a file encoding from text, using either the chardet lib or by trying to decode the file. fallback detection (text) Simple detection based on BOM in case chardet is not available. findid(id) find unit with matching id by checking id\_index findunit (source) Find the unit with the given source string. **Return type** TranslationUnit or None findunits (source) Find the units with the given source string. Return type TranslationUnit or None getids (filename=None) return a list of unit ids getprojectstyle() Get the project type for this store. getsourcelanguage() Get the source language for this store. gettargetlanguage() Get the target language for this store. getunits() Return a list of all units in this store. isempty() Return True if the object doesn't contain any translation units. makeindex()

**Returns** The default matching criterion for all the subclasses. **Return type** string

Indexes the items in this store. At least .sourceindex should be useful.

The matching criterion to use when merging on.

```
parse (input)
          parsese the given file or file source string
     classmethod parsefile(storefile)
          Reads the given file (or opens the given filename) and parses back to an object.
     classmethod parsestring(storestring)
          Convert the string representation back to an object.
     remove unit from index(unit)
          Remove a unit from source and locaton indexes
     require_index()
          make sure source index exists
     save()
          Save to the file that data was originally read from, if available.
     savefile (storefile)
          Write the string representation to the given file (or filename).
     serialize (out)
          Converts to a bytes representation that can be parsed back using parsestring (). out should be an
          open file-like objects to write to.
     setprojectstyle (project_style)
          Set the project type for this store.
     setsourcelanguage (sourcelanguage)
          Set the source language for this store.
     settargetlanguage (targetlanguage)
          Set the target language for this store.
     translate(source)
          Return the translated string for a given source string.
              Return type String or None
     unit_iter()
          Iterator over all the units in this store.
class translate.storage.wordfast.WordfastTime(newtime=None)
     Manages time stamps in the Wordfast format of YYYYMMDD~hhmmss
     get_time()
          Get the time_struct object
     get timestring()
          Get the time in the Wordfast time format
     set_time (newtime)
          Set the time_struct object
              Parameters newtime (time.time_struct) - a new time object
     set_timestring(timestring)
          Set the time_sturct object using a Wordfast time formated string
              Parameters timestring (String) – A Wordfast time string (YYYMMDD~hhmmss)
     time
          Get the time struct object
```

### timestring

Get the time in the Wordfast time format

### class translate.storage.wordfast.WordfastUnit(source=None)

A Wordfast translation memory unit

#### adderror (errorname, errortext)

Adds an error message to this unit.

#### **Parameters**

- **errorname** (*string*) A single word to id the error.
- **errortext** (*string*) The text describing the error.

#### addlocation (location)

Add one location to the list of locations.

**Note:** Shouldn't be implemented if the format doesn't support it.

### addlocations (location)

Add a location or a list of locations.

**Note:** Most classes shouldn't need to implement this, but should rather implement TranslationUnit.addlocation().

**Warning:** This method might be removed in future.

# addnote (text, origin=None, position='append')

Adds a note (comment).

### **Parameters**

- **text** (*string*) Usually just a sentence or two.
- **origin** (*string*) Specifies who/where the comment comes from. Origin can be one of the following text strings: 'translator' 'developer', 'programmer', 'source code' (synonyms)

### classmethod buildfromunit (unit)

Build a native unit from a foreign unit, preserving as much information as possible.

### dict

Get the dictionary of values for a Wordfast line

#### getcontext()

Get the message context.

# getdict()

Get the dictionary of values for a Wordfast line

### geterrors()

Get all error messages.

### **Return type** Dictionary

### getid()

A unique identifier for this unit.

### Return type string

Returns an identifier for this unit that is unique in the store

Derived classes should override this in a way that guarantees a unique identifier for each unit in the store.

### getlocations()

A list of source code locations.

#### Return type List

Note: Shouldn't be implemented if the format doesn't support it.

### getnotes (origin=None)

Returns all notes about this unit.

It will probably be freeform text or something reasonable that can be synthesised by the format. It should not include location comments (see getlocations()).

### gettargetlen()

Returns the length of the target string.

### Return type Integer

Note: Plural forms might be combined.

#### getunits()

This unit in a list.

# hasplural()

Tells whether or not this specific unit has plural strings.

### infer\_state()

Empty method that should be overridden in sub-classes to infer the current state(\_n) of the unit from its current state.

### isblank()

Used to see if this unit has no source or target string.

**Note:** This is probably used more to find translatable units, and we might want to move in that direction rather and get rid of this.

#### isfuzzy()

Indicates whether this unit is fuzzy.

### isheader()

Indicates whether this unit is a header.

# $\verb|isobsolete|()|$

indicate whether a unit is obsolete

#### isreview()

Indicates whether this unit needs review.

# istranslatable()

Indicates whether this unit can be translated.

This should be used to distinguish real units for translation from header, obsolete, binary or other blank units.

### istranslated()

Indicates whether this unit is translated.

This should be used rather than deducing it from .target, to ensure that other classes can implement more functionality (as XLIFF does).

#### makeobsolete()

Make a unit obsolete

# markfuzzy(value=True)

Marks the unit as fuzzy or not.

### markreviewneeded (needsreview=True, explanation=None)

Marks the unit to indicate whether it needs review.

#### **Parameters**

- needsreview Defaults to True.
- **explanation** Adds an optional explanation as a note.

```
merge (otherunit, overwrite=False, comments=True, authoritative=False)
```

Do basic format agnostic merging.

### multistring\_to\_rich (mulstring)

Convert a multistring to a list of "rich" string trees:

#### removenotes (origin=None)

Remove all the translator's notes.

# rich\_source

#### See also:

```
rich_to_multistring(), multistring_to_rich()
```

### rich\_target

#### See also:

```
rich to multistring(), multistring to rich()
```

### classmethod rich\_to\_multistring(elem\_list)

Convert a "rich" string tree to a multistring:

```
>>> from translate.storage.placeables.interfaces import X
>>> rich = [StringElem(['foo', X(id='xxx', sub=[' ']), 'bar'])]
>>> TranslationUnit.rich_to_multistring(rich)
multistring(u'foo bar')
```

#### setcontext (context)

Set the message context

Iterator that only returns this unit.

```
setdict (newdict)
    Set the dictionary of values for a Wordfast line
    Parameters newdict (Dict) - a new dictionary with Wordfast line elements
setid (value)
    Sets the unique identified for this unit.
    only implemented if format allows ids independant from other unit properties like source or context
unit_iter()
```

#### workflow

A workflow is defined by a set of states that a translation unit can be in and the (allowed) transitions between these states. A state is defined by a range between -128 and 127, indicating its level of "completeness". The range is closed at the beginning and open at the end. That is, if a workflow contains states A, B and C where A < B < C, a unit with state number n is in state A if  $A \le n < B$ , state B if  $B \le n < C$  or state C if  $C \le n < C$ .

A value of 0 is typically the "empty" or "new" state with negative values reserved for states like "obsolete" or "do not use".

Format specific workflows should be defined in such a way that the numeric state values correspond to similar states. For example state 0 should be "untranslated" in PO and "new" or "empty" in XLIFF, state 100 should be "translated" in PO and "final" in XLIFF. This allows formats to implicitly define similar states.

```
exception translate.storage.workflow.InvalidStateObjectError (obj)
exception translate.storage.workflow.NoInitialStateError
class translate.storage.workflow.StateEnum
    Only contains the constants for default states.
exception translate.storage.workflow.StateNotInWorkflowError(state)
exception translate.storage.workflow.TransitionError
exception translate.storage.workflow.WorkflowError
xliff
Module for handling XLIFF files for translation.
The official recommendation is to use the extention .xlf for XLIFF files.
class translate.storage.xliff.xlifffile(*args, **kwargs)
    Class representing a XLIFF file store.
    UnitClass
         alias of xliffunit
    add_unit_to_index(unit)
         Add a unit to source and location idexes
    addheader()
         Initialise the file header.
    addsourceunit (source, filename='NoName', createifmissing=False)
```

instead (will create the nodes required if asked). Returns success

adds the given trans-unit to the last used body node if the filename has changed it uses the slow method

```
addunit (unit, new=True)
     Append the given unit to the object's list of units.
     This method should always be used rather than trying to modify the list manually.
         Parameters unit (TranslationUnit) - The unit that will be added.
createfilenode (filename, sourcelanguage=None, targetlanguage=None, datatype='plaintext')
     creates a filenode with the given filename. All parameters are needed for XLIFF compliance.
creategroup (filename='NoName', createifmissing=False, restype=None)
     adds a group tag into the specified file
detect_encoding(text, default_encodings=None)
     Try to detect a file encoding from text, using either the chardet lib or by trying to decode the file.
fallback_detection (text)
     Simple detection based on BOM in case chardet is not available.
findid(id)
     find unit with matching id by checking id_index
findunit (source)
     Find the unit with the given source string.
         Return type TranslationUnit or None
findunits (source)
     Find the units with the given source string.
         Return type TranslationUnit or None
getbodynode (filenode, createifmissing=False)
     finds the body node for the given filenode
getdatatype (filename=None)
     Returns the datatype of the stored file. If no filename is given, the datatype of the first file is given.
getdate (filename=None)
     Returns the date attribute for the file.
     If no filename is given, the date of the first file is given. If the date attribute is not specified, None is
     returned.
         Returns Date attribute of file
         Return type Date or None
getfilename (filenode)
     returns the name of the given file
getfilenames()
     returns all filenames in this XLIFF file
getfilenode (filename, createifmissing=False)
     finds the filenode with the given name
getheadernode (filenode, createifmissing=False)
     finds the header node for the given filenode
getids (filename=None)
     return a list of unit ids
```

4.1. API 673

getprojectstyle()

Get the project type for this store.

#### getsourcelanguage()

Get the source language for this store.

### gettargetlanguage()

Get the target language for this store.

#### getunits()

Return a list of all units in this store.

#### initbody()

Initialises self.body so it never needs to be retrieved from the XML again.

# isempty()

Return True if the object doesn't contain any translation units.

#### makeindex()

Indexes the items in this store. At least .sourceindex should be useful.

#### merge\_on

The matching criterion to use when merging on.

**Returns** The default matching criterion for all the subclasses.

**Return type** string

# namespaced (name)

Returns name in Clark notation.

For example namespaced ("source") in an XLIFF document might return:

```
{urn:oasis:names:tc:xliff:document:1.1}source
```

This is needed throughout lxml.

### parse(xml)

Populates this object from the given xml string

### classmethod parsefile(storefile)

Reads the given file (or opens the given filename) and parses back to an object.

### classmethod parsestring(storestring)

Parses the string to return the correct file object

# ${\tt remove\_unit\_from\_index}~(\textit{unit})$

Remove a unit from source and locaton indexes

#### removedefaultfile()

We want to remove the default file-tag as soon as possible if we know if still present and empty.

### require\_index()

make sure source index exists

### save()

Save to the file that data was originally read from, if available.

# ${\tt savefile}\ (storefile)$

Write the string representation to the given file (or filename).

### serialize(out)

Converts to a string containing the file's XML

### setfilename (filenode, filename)

set the name of the given file

### setprojectstyle (project\_style)

Set the project type for this store.

# setsourcelanguage(language)

Set the source language for this store.

### settargetlanguage (language)

Set the target language for this store.

#### suggestions in format = True

xliff units have alttrans tags which can be used to store suggestions

# switchfile (filename, createifmissing=False)

Adds the given trans-unit (will create the nodes required if asked).

**Returns** Success

Return type Boolean

### translate(source)

Return the translated string for a given source string.

Return type String or None

### unit\_iter()

Iterator over all the units in this store.

# class translate.storage.xliff.xliffunit (source, empty=False, \*\*kwargs)

A single term in the xliff file.

# $\textbf{addalttrans} \ (\textit{txt}, \textit{origin} = None, \textit{lang} = None, \textit{sourcetxt} = None, \textit{matchquality} = None)$

Adds an alt-trans tag and alt-trans components to the unit.

**Parameters** txt (String) – Alternative translation of the source text.

# adderror (errorname, errortext)

Adds an error message to this unit.

### addlocation (location)

Add one location to the list of locations.

**Note:** Shouldn't be implemented if the format doesn't support it.

### addlocations (location)

Add a location or a list of locations.

**Note:** Most classes shouldn't need to implement this, but should rather implement TranslationUnit.addlocation().

Warning: This method might be removed in future.

# addnote (text, origin=None, position='append')

Add a note specifically in a "note" tag

#### classmethod buildfromunit (unit)

Build a native unit from a foreign unit, preserving as much information as possible.

#### correctorigin (node, origin)

Check against node tag's origin (e.g note or alt-trans)

### createcontextgroup (name, contexts=None, purpose=None)

Add the context group to the trans-unit with contexts a list with (type, text) tuples describing each context.

### createlanguageNode (lang, text, purpose)

Returns an xml Element setup with given parameters.

#### delalttrans (alternative)

Removes the supplied alternative from the list of alt-trans tags

# getNodeText (languageNode, xml\_space='preserve')

Retrieves the term from the given languageNode.

#### get\_rich\_target (lang=None)

retrieves the "target" text (second entry), or the entry in the specified language, if it exists

#### getalttrans(origin=None)

Returns <alt-trans> for the given origin as a list of units. No origin means all alternatives.

#### getcontext()

Get the message context.

### getcontextgroups (name)

Returns the contexts in the context groups with the specified name

#### geterrors()

Get all error messages.

#### getid()

A unique identifier for this unit.

### Return type string

**Returns** an identifier for this unit that is unique in the store

Derived classes should override this in a way that guarantees a unique identifier for each unit in the store.

#### getlanguageNode (lang=None, index=None)

Retrieves a languageNode either by language or by index.

### getlanguageNodes()

We override this to get source and target nodes.

# getlocations()

A list of source code locations.

### **Return type** List

**Note:** Shouldn't be implemented if the format doesn't support it.

# getnotes (origin=None)

Returns all notes about this unit.

It will probably be freeform text or something reasonable that can be synthesised by the format. It should not include location comments (see getlocations()).

### getrestype()

returns the restype attribute in the trans-unit tag

# gettarget (lang=None)

retrieves the "target" text (second entry), or the entry in the specified language, if it exists

#### gettargetlen()

Returns the length of the target string.

### Return type Integer

Note: Plural forms might be combined.

#### getunits()

This unit in a list.

### hasplural()

Tells whether or not this specific unit has plural strings.

#### infer\_state()

Empty method that should be overridden in sub-classes to infer the current state(\_n) of the unit from its current state.

### isapproved()

States whether this unit is approved.

#### isblank()

Used to see if this unit has no source or target string.

**Note:** This is probably used more to find translatable units, and we might want to move in that direction rather and get rid of this.

### isfuzzy()

Indicates whether this unit is fuzzy.

### isheader()

Indicates whether this unit is a header.

#### isobsolete()

indicate whether a unit is obsolete

### isreview()

States whether this unit needs to be reviewed

### istranslatable()

Indicates whether this unit can be translated.

This should be used to distinguish real units for translation from header, obsolete, binary or other blank units.

### istranslated()

Indicates whether this unit is translated.

This should be used rather than deducing it from .target, to ensure that other classes can implement more functionality (as XLIFF does).

### makeobsolete()

Make a unit obsolete

### markapproved(value=True)

Mark this unit as approved.

### markfuzzy(value=True)

Marks the unit as fuzzy or not.

#### markreviewneeded (needsreview=True, explanation=None)

Marks the unit to indicate whether it needs review.

Adds an optional explanation as a note.

### merge (otherunit, overwrite=False, comments=True, authoritative=False)

Do basic format agnostic merging.

### classmethod multistring\_to\_rich(mstr)

Override TranslationUnit.multistring\_to\_rich() which is used by the rich\_source and rich\_target properties.

### namespaced(name)

Returns name in Clark notation.

For example namespaced ("source") in an XLIFF document might return:

```
{urn:oasis:names:tc:xliff:document:1.1}source
```

This is needed throughout lxml.

### removenotes (origin=None)

Remove all the translator notes.

# classmethod rich\_to\_multistring(elem\_list)

Override TranslationUnit.rich\_to\_multistring() which is used by the rich\_source and rich\_target properties.

#### setcontext (context)

Set the message context

### setid(id)

Sets the unique identified for this unit.

only implemented if format allows ids independant from other unit properties like source or context

```
settarget (target, lang='xx', append=False)
```

Sets the target string to the given value.

#### unit\_iter()

Iterator that only returns this unit.

#### xml extract

#### extract

 $nsmap=\{\}$ )

Maintain constants and variables used during the walking of a DOM tree (via the function apply).

A node corresponds to a translatable element. A node may have children, which correspond to placeables.

### has\_translatable\_text

Check if it contains any chunk of text with more than whitespace.

If not, then there's nothing to translate.

```
\label{lem:condition} \verb|translate.storage.xml_extract.extract.build_idml_store| (odf_file, store, parse_state, store_adder=None) \\ Build a store for the given IDML file.
```

Build a store for the given XML file.

```
translate.storage.xml_extract.extract.make_postore_adder(store, id_maker, file-name)
```

Return a function which, when called with a Translatable will add a unit to 'store'. The placeables will be represented as strings according to 'placeable quoter'.

```
translate.storage.xml_extract.extract.process_translatable (dom_node, state)
Process a translatable DOM node.
```

Any translatable content present in a child node is treated as a placeable.

### generate

```
translate.storage.xml_extract.generate.find_dom_root(parent_dom_node, dom_node)
```

#### See also:

```
find_placeable_dom_tree_roots()
```

```
translate.storage.xml_extract.generate.find_placeable_dom_tree_roots (unit_node) For an inline placeable, find the root DOM node for the placeable in its parent.
```

Consider the diagram. In this pseudo-ODF example, there is an inline span element. However, the span is contained in other tags (which we never process). When splicing the template DOM tree (that is, the DOM which comes from the XML document we're using to generate a translated XML document), we'll need to move DOM sub-trees around and we need the roots of these sub-trees:

```
translate.storage.xml_extract.generate.get_xliff_source_target_doms (unit) Return a tuple with unit source and target DOM objects.
```

This method is method is meant to provide a way to retrieve the DOM objects for the unit source and target for XLIFF stores.

Return a function:

```
action: etree_Element x base.TranslationUnit -> None
```

which takes a dom\_node and a translation unit. The dom\_node is rearranged according to rearrangement of placeables in unit.target (relative to their positions in unit.source).

#### misc

```
translate.storage.xml_extract.misc.compose_mappings(left, right)
```

Given two mappings left: A -> B and right: B -> C, create a hash result\_map: A -> C. Only values in left (i.e. things from B) which have corresponding keys in right will have their keys mapped to values in right.

translate.storage.xml\_extract.misc.parse\_tag(full\_tag)

```
translate.storage.xml_extract.misc.reduce_tree(f, parent_unit_node, unit_node, get_children, *state)
```

Enumerate a tree, applying f to in a pre-order fashion to each node.

parent\_unit\_node contains the parent of unit\_node. For the root of the tree, parent\_unit\_node == unit\_node.

get\_children is a single argument function applied to a unit\_node to get a list/iterator to its children.

state is used by f to modify state information relating to whatever f does to the tree.

# unit tree

```
translate.storage.xml_extract.unit_tree.build_unit_tree(store, filename=None)
```

Enumerate a translation store and build a tree with XPath components as nodes and where a node contains a unit if a path from the root of the tree to the node containing the unit, is equal to the XPath of the unit.

The tree looks something like this:

```
root
'- ('document-content', 1)
'- ('body', 2)
|- ('text', 1)
| '- ('p', 1)
| '- <reference to a unit>
|- ('text', 2)
| '- ('p', 1)
| '- <reference to a unit>
'- ('text', 3)
'- ('p', 1)
| '- <reference to a unit>
```

# xpath\_breadcrumb

```
class translate.storage.xml_extract.xpath_breadcrumb.XPathBreadcrumb
```

A class which is used to build XPath-like paths as a DOM tree is walked. It keeps track of the number of times which it has seen a certain tag, so that it will correctly create indices for tags.

Initially, the path is empty. Thus >>> xb = XPathBreadcrumb() >>> xb.xpath ""

Suppose we walk down a DOM node for the tag <foo> and we want to record this, we simply do >>> xb.start\_tag('foo')

Now, the path is no longer empty. Thus >>> xb.xpath foo[0]

Now suppose there are two <bar> tags under the tag <foo> (that is <foo> <bar> </bar> <bar> <br/> then the breadcrumb will keep track of the number of times it sees <br/> <br/> Thus

```
>>> xb.start_tag('bar')
>>> xb.xpath
foo[0]/bar[0]
>>> xb.end_tag()
>>> xb.xpath
foo[0]
>>> xb.start_tag('bar')
>>> xb.start_tag('bar')
```

#### xml name

```
class translate.storage.xml_name.XmlNamer(dom_node)
```

Initialize me with a DOM node or a DOM document node (the toplevel node you get when parsing an XML file). Then use me to generate fully qualified XML names.

I can also give you XmlNamespace objects if you give me the abbreviated namespace name. These are useful if you need to reference a namespace continuously.

```
>>> office_ns = name.namespace('office')
>>> office_ns.name('foo')
{urn:oasis:names:tc:opendocument:xmlns:office:1.0}foo
```

#### zip

This module provides functionality to work with zip files.

```
class translate.storage.zip.ZIPFile (filename=None)
    This class represents a ZIP file like a directory.

file_iter()
    Iterator over (dir, filename) for all files in this directory.

getfiles()
    Returns a list of (dir, filename) tuples for all the file names in this directory.

getunits()
    List of all the units in all the files in this directory.

scanfiles()
    Populate the internal file data.
```

4.1. API 681

```
unit iter()
```

Iterator over all the units in all the files in this zip file.

#### tools

Code to perform various operations, mostly on po files.

## build\_tmdb

Import units from translations files into tmdb.

## phppo2pypo

Convert PHP format .po files to Python format .po files.

```
translate.tools.phppo2pypo.convertphp2py (inputfile, outputfile, template=None)
Converts from PHP .po format to Python .po format
```

#### **Parameters**

- inputfile file handle of the source
- outputfile file handle to write to
- template unused

```
translate.tools.phppo2pypo.main(argv=None)
Converts PHP.po files to Python.po files.
```

#### poclean

Produces a clean file from an unclean file (Trados/Wordfast) by stripping out the tw4win indicators.

This does not convert an RTF file to PO/XLIFF, but produces the target file with only the target text in from a text version of the RTF.

```
translate.tools.poclean.cleanfile(thefile)
    cleans the given file

translate.tools.poclean.cleanunit(unit)
    cleans the targets in the given unit

translate.tools.poclean.runclean(inputfile, outputfile, templatefile)
    reads in inputfile, cleans, writes to outputfile
```

## pocompile

Compile XLIFF and Gettext PO localization files into Gettext MO (Machine Object) files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/pocompile.html for examples and usage instructions.

```
translate.tools.pocompile.convertmo (inputfile, outputfile, templatefile, includefuzzy=False) reads in a base class derived inputfile, converts using pocompile, writes to outputfile
```

## poconflicts

Conflict finder for Gettext PO localization files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/poconflicts.html for examples and usage instructions.

Check that the supplied option values and leftover arguments are valid. Returns the option values and leftover arguments (possibly adjusted, possibly completely new – whatever you like). Default implementation just returns the passed-in values; subclasses may override as desired.

#### checkoutputsubdir(options, subdir)

Checks to see if subdir under options.output needs to be created, creates if neccessary.

```
clean (string, options)
```

returns the cleaned string that contains the text to be matched

```
define option (option)
```

Defines the given option, replacing an existing one of the same short name if neccessary...

## destroy()

Declare that you are done with this OptionParser. This cleans up reference cycles so the OptionParser (and all objects referenced by it) can be garbage-collected promptly. After calling destroy(), the OptionParser is unusable.

#### disable\_interspersed\_args()

Set parsing to stop on the first non-option. Use this if you have a command processor which runs another command that has options of its own and you want to make sure these options don't get confused.

## enable\_interspersed\_args()

Set parsing to not stop on the first non-option, allowing interspersing switches with command arguments. This is the default behavior. See also disable\_interspersed\_args() and the class documentation description of the attribute allow interspersed args.

```
error (msg : string)
```

Print a usage message incorporating 'msg' to stderr and exit. If you override this in a subclass, it should not return – it should either exit or raise an exception.

## finalizetempoutputfile (options, outputfile, fulloutputpath)

Write the temp outputfile to its final destination.

## flatten (text, joinchar)

flattens text to just be words

#### format\_manpage()

returns a formatted manpage

4.1. API 683

```
getformathelp(formats)
     Make a nice help string for describing formats...
getfullinputpath (options, inputpath)
     Gets the absolute path to an input file.
getfulloutputpath (options, outputpath)
     Gets the absolute path to an output file.
getfulltemplatepath (options, templatepath)
     Gets the absolute path to a template file.
getoutputname (options, inputname, outputformat)
     Gets an output filename based on the input filename.
getoutputoptions (options, inputpath, templatepath)
     Works out which output format and processor method to use...
getpassthroughoptions(options)
     Get the options required to pass to the filtermethod...
gettemplatename (options, inputname)
     Gets an output filename based on the input filename.
getusageman (option)
     returns the usage string for the given option
getusagestring(option)
     returns the usage string for the given option
isexcluded(options, inputpath)
     Checks if this path has been excluded.
isrecursive (fileoption, filepurpose='input')
     Checks if fileoption is a recursive file.
isvalidinputname (inputname)
     Checks if this is a valid input filename.
mkdir (parent, subdir)
     Makes a subdirectory (recursively if neccessary).
openinputfile (options, fullinputpath)
     Opens the input file.
openoutputfile (options, fulloutputpath)
     Opens the output file.
opentemplatefile (options, fulltemplatepath)
     Opens the template file (if required).
opentempoutputfile (options, fulloutputpath)
     Opens a temporary output file.
outputconflicts(options)
     saves the result of the conflict match
parse_args (args=None, values=None)
     parses the command line options, handling implicit input/output args
print_help (file : file = stdout)
     Print an extended help message, listing all options and any help text provided with them, to 'file' (default
```

stdout).

#### print manpage(file=None)

outputs a manpage for the program using the help information

#### print\_usage (file : file = stdout)

Print the usage message for the current program (self.usage) to 'file' (default stdout). Any occurrence of the string "%prog" in self.usage is replaced with the name of the current program (basename of sys.argv[0]). Does nothing if self.usage is empty or not defined.

#### print\_version (file : file = stdout)

Print the version message for this program (self.version) to 'file' (default stdout). As with print\_usage(), any occurrence of "%prog" in self.version is replaced by the current program's name. Does nothing if self.version is empty or undefined.

#### processfile (fileprocessor, options, fullinputpath)

process an individual file

## recurseinputfilelist(options)

Use a list of files, and find a common base directory for them.

#### recurseinputfiles (options)

Recurse through directories and return files to be processed.

#### recursiveprocess (options)

recurse through directories and process files

#### run()

Parses the arguments, and runs recursive process with the resulting options...

## set\_usage (usage=None)

sets the usage string - if usage not given, uses getusagestring for each option

#### seterrorleveloptions()

Sets the errorlevel options.

## setformats (formats, usetemplates)

Sets the format options using the given format dictionary.

**Parameters formats** (Dictionary) – The dictionary keys should be:

- Single strings (or 1-tuples) containing an input format (if not *usetemplates*)
- Tuples containing an input format and template format (if usetemplates)
- Formats can be *None* to indicate what to do with standard input

The dictionary values should be tuples of outputformat (string) and processor method.

#### setmanpageoption()

creates a manpage option that allows the optionparser to generate a manpage

#### setprogressoptions()

Sets the progress options.

## splitext (pathname)

Splits *pathname* into name and ext, and removes the extsep.

Parameters pathname (string) - A file path

Returns root, ext

Return type tuple

## splitinputext(inputpath)

Splits an *inputpath* into name and extension.

4.1. API 685

```
splittemplateext (templatepath)
```

Splits a *templatepath* into name and extension.

```
templateexists (options, templatepath)
```

Returns whether the given template exists...

```
warning (msg, options=None, exc_info=None)
```

Print a warning message incorporating 'msg' to stderr and exit.

#### pocount

Count strings and words for supported localization files.

These include: XLIFF, TMX, Gettex PO and MO, Qt .ts and .qm, Wordfast TM, etc

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/pocount.html for examples and usage instructions.

```
class translate.tools.pocount.ConsoleColor
```

Class to implement color mode.

```
translate.tools.pocount.calcstats_old(filename)
```

This is the previous implementation of calcstats() and is left for comparison and debuging purposes.

translate.tools.pocount.**summarize** (*title*, *stats*, *style=0*, *indent=8*, *incomplete\_only=False*)

Print summary for a .po file in specified format.

#### **Parameters**

- title name of .po file
- **stats** array with translation statistics for the file specified
- **indent** indentation of the 2nd column (length of longest filename)
- incomplete\_only (Boolean) omit fully translated files

Return type Boolean

**Returns** 1 if counting incomplete files (incomplete\_only=True) and the file is completely translated, 0 otherwise

#### podebug

Insert debug messages into XLIFF and Gettext PO localization files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/podebug.html for examples and usage instructions.

```
translate. tools. podebug. \begin{tabular}{ll} convertpo (inputfile, & outputfile, & template file, & format=None, \\ rewrite style=None, & ignore option=None, & preserve place-holders=None) \\ \end{tabular}
```

Reads in inputfile, changes it to have debug strings, writes to outputfile.

## pogrep

Grep XLIFF, Gettext PO and TMX localization files.

Matches are output to snippet files of the same type which can then be reviewed and later merged using *pomerge*.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/pogrep.html for examples and usage instructions.

```
class translate.tools.pogrep.GrepMatch (unit, part='target', part_n=0, start=0, end=0)

Just a small data structure that represents a search match.
```

```
class translate.tools.pogrep.GrepOptionParser(formats, usetemplates=False, allowmiss-
ingtemplate=False, description=None)
```

a specialized Option Parser for the grep tool...

```
add_option(Option)
    add_option(opt_str, ..., kwarg=val, ...)
check_values(values: Values, args: [string])
```

-> (values : Values, args : [string])

Check that the supplied option values and leftover arguments are valid. Returns the option values and leftover arguments (possibly adjusted, possibly completely new – whatever you like). Default implementation just returns the passed-in values; subclasses may override as desired.

## checkoutputsubdir(options, subdir)

Checks to see if subdir under options.output needs to be created, creates if neccessary.

#### define option(option)

Defines the given option, replacing an existing one of the same short name if neccessary...

## destroy()

Declare that you are done with this OptionParser. This cleans up reference cycles so the OptionParser (and all objects referenced by it) can be garbage-collected promptly. After calling destroy(), the OptionParser is unusable.

## disable\_interspersed\_args()

Set parsing to stop on the first non-option. Use this if you have a command processor which runs another command that has options of its own and you want to make sure these options don't get confused.

## enable\_interspersed\_args()

Set parsing to not stop on the first non-option, allowing interspersing switches with command arguments. This is the default behavior. See also disable\_interspersed\_args() and the class documentation description of the attribute allow\_interspersed\_args.

```
error (msg : string)
```

Print a usage message incorporating 'msg' to stderr and exit. If you override this in a subclass, it should not return – it should either exit or raise an exception.

## finalizetempoutputfile (options, outputfile, fulloutputpath)

Write the temp outputfile to its final destination.

#### format\_manpage()

returns a formatted manpage

#### getformathelp (formats)

Make a nice help string for describing formats...

## getfullinputpath (options, inputpath)

Gets the absolute path to an input file.

#### getfulloutputpath (options, outputpath)

Gets the absolute path to an output file.

#### getfulltemplatepath (options, templatepath)

Gets the absolute path to a template file.

4.1. API 687

## getoutputname (options, inputname, outputformat) Gets an output filename based on the input filename. getoutputoptions (options, inputpath, templatepath) Works out which output format and processor method to use... getpassthroughoptions (options) Get the options required to pass to the filtermethod... gettemplatename (options, inputname) Gets an output filename based on the input filename. getusageman (option) returns the usage string for the given option getusagestring(option) returns the usage string for the given option isexcluded(options, inputpath) Checks if this path has been excluded. isrecursive (fileoption, filepurpose='input') Checks if fileoption is a recursive file. isvalidinputname (inputname) Checks if this is a valid input filename. **mkdir** (parent, subdir) Makes a subdirectory (recursively if neccessary). openinputfile (options, fullinputpath) Opens the input file. openoutputfile (options, fulloutputpath) Opens the output file. opentemplatefile (options, fulltemplatepath) Opens the template file (if required). opentempoutputfile (options, fulloutputpath) Opens a temporary output file. parse\_args (args=None, values=None) parses the command line options, handling implicit input/output args print\_help (file : file = stdout) Print an extended help message, listing all options and any help text provided with them, to 'file' (default stdout).

## print\_manpage (file=None)

outputs a manpage for the program using the help information

```
print_usage (file : file = stdout)
```

Print the usage message for the current program (self.usage) to 'file' (default stdout). Any occurrence of the string "%prog" in self.usage is replaced with the name of the current program (basename of sys.argv[0]). Does nothing if self.usage is empty or not defined.

#### print\_version (file : file = stdout)

Print the version message for this program (self.version) to 'file' (default stdout). As with print\_usage(), any occurrence of "%prog" in self.version is replaced by the current program's name. Does nothing if self.version is empty or undefined.

```
processfile (fileprocessor, options, fullinputpath, fulloutputpath, fulltemplatepath)
           Process an individual file.
     recurseinputfilelist (options)
           Use a list of files, and find a common base directory for them.
     recurseinputfiles (options)
           Recurse through directories and return files to be processed.
     recursiveprocess (options)
           Recurse through directories and process files.
     run()
           parses the arguments, and runs recursive process with the resulting options
     set_usage (usage=None)
           sets the usage string - if usage not given, uses getusagestring for each option
     seterrorleveloptions()
           Sets the errorlevel options.
     setformats (formats, usetemplates)
           Sets the format options using the given format dictionary.
               Parameters formats (Dictionary) – The dictionary keys should be:
                   • Single strings (or 1-tuples) containing an input format (if not usetemplates)
                   • Tuples containing an input format and template format (if usetemplates)
                   • Formats can be None to indicate what to do with standard input
                   The dictionary values should be tuples of outputformat (string) and processor method.
     setmanpageoption()
           creates a manpage option that allows the optionparser to generate a manpage
     setprogressoptions()
           Sets the progress options.
     splitext (pathname)
           Splits pathname into name and ext, and removes the extsep.
               Parameters pathname (string) - A file path
               Returns root, ext
               Return type tuple
     splitinputext(inputpath)
           Splits an inputpath into name and extension.
     splittemplateext(templatepath)
           Splits a templatepath into name and extension.
     templateexists (options, templatepath)
           Returns whether the given template exists...
     warning (msg, options=None, exc_info=None)
           Print a warning message incorporating 'msg' to stderr and exit.
translate.tools.pogrep.find_matches(unit, part, strings, re_search)
```

4.1. API 689

Return the GrepFilter objects where re\_search matches in strings.

```
translate.tools.pogrep.real_index(string, nfc_index)
```

Calculate the real index in the unnormalized string that corresponds to the index nfc\_index in the normalized string.

```
translate.tools.pogrep.rungrep(inputfile, outputfile, templatefile, checkfilter) reads in inputfile, filters using checkfilter, writes to outputfile
```

#### pomerge

Merges XLIFF and Gettext PO localization files.

Return type Boolean

Snippet file produced by e.g. pogrep and updated by a translator can be merged back into the original files.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/pomerge.html for examples and usage instructions.

```
translate.tools.pomerge.mergestores(store1, store2, mergeblanks, mergefuzzy, mergecomments)
Take any new translations in store2 and write them into store1.

translate.tools.pomerge.str2bool(option)
Convert a string value to boolean

Parameters option(String) - yes, true, 1, no, false, 0
```

#### porestructure

Restructure Gettxt PO files produced by *poconflicts* into the original directory tree for merging using *pomerge*.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/pomerge.html for examples and usage instructions.

Check that the supplied option values and leftover arguments are valid. Returns the option values and leftover arguments (possibly adjusted, possibly completely new – whatever you like). Default implementation just returns the passed-in values; subclasses may override as desired.

```
checkoutputsubdir(options, subdir)
```

Checks to see if subdir under options.output needs to be created, creates if neccessary.

```
define_option(option)
```

Defines the given option, replacing an existing one of the same short name if neccessary...

#### destroy()

Declare that you are done with this OptionParser. This cleans up reference cycles so the OptionParser (and all objects referenced by it) can be garbage-collected promptly. After calling destroy(), the OptionParser is unusable.

#### disable\_interspersed\_args()

Set parsing to stop on the first non-option. Use this if you have a command processor which runs another command that has options of its own and you want to make sure these options don't get confused.

#### enable\_interspersed\_args()

Set parsing to not stop on the first non-option, allowing interspersing switches with command arguments. This is the default behavior. See also disable\_interspersed\_args() and the class documentation description of the attribute allow\_interspersed\_args.

## error (msg : string)

Print a usage message incorporating 'msg' to stderr and exit. If you override this in a subclass, it should not return – it should either exit or raise an exception.

#### finalizetempoutputfile (options, outputfile, fulloutputpath)

Write the temp outputfile to its final destination.

#### format\_manpage()

returns a formatted manpage

#### getformathelp (formats)

Make a nice help string for describing formats...

#### getfullinputpath (options, inputpath)

Gets the absolute path to an input file.

## getfulloutputpath (options, outputpath)

Gets the absolute path to an output file.

#### getfulltemplatepath (options, templatepath)

Gets the absolute path to a template file.

#### getoutputname (options, inputname, outputformat)

Gets an output filename based on the input filename.

## getoutputoptions (options, inputpath, templatepath)

Works out which output format and processor method to use...

## getpassthroughoptions(options)

Get the options required to pass to the filtermethod...

#### gettemplatename (options, inputname)

Gets an output filename based on the input filename.

## getusageman (option)

returns the usage string for the given option

#### getusagestring(option)

returns the usage string for the given option

#### isexcluded(options, inputpath)

Checks if this path has been excluded.

## isrecursive (fileoption, filepurpose='input')

Checks if fileoption is a recursive file.

#### isvalidinputname (inputname)

Checks if this is a valid input filename.

#### mkdir (parent, subdir)

Makes a subdirectory (recursively if neccessary).

## ${\tt openinputfile} \ (options, full input path)$

Opens the input file.

4.1. API 691

#### openoutputfile (options, fulloutputpath)

Opens the output file.

#### opentemplatefile (options, fulltemplatepath)

Opens the template file (if required).

## opentempoutputfile (options, fulloutputpath)

Opens a temporary output file.

#### parse args (args=None, values=None)

parses the command line options, handling implicit input/output args

#### print\_help (file : file = stdout)

Print an extended help message, listing all options and any help text provided with them, to 'file' (default stdout).

#### print\_manpage (file=None)

outputs a manpage for the program using the help information

#### print\_usage (file : file = stdout)

Print the usage message for the current program (self.usage) to 'file' (default stdout). Any occurrence of the string "%prog" in self.usage is replaced with the name of the current program (basename of sys.argv[0]). Does nothing if self.usage is empty or not defined.

#### print\_version (file : file = stdout)

Print the version message for this program (self.version) to 'file' (default stdout). As with print\_usage(), any occurrence of "%prog" in self.version is replaced by the current program's name. Does nothing if self.version is empty or undefined.

#### processfile (options, fullinputpath)

process an individual file

## recurseinputfilelist (options)

Use a list of files, and find a common base directory for them.

#### recurseinputfiles (options)

Recurse through directories and return files to be processed.

#### recursiveprocess (options)

recurse through directories and process files

## run()

Parses the arguments, and runs recursive process with the resulting options...

## set\_usage (usage=None)

sets the usage string - if usage not given, uses getusagestring for each option

#### seterrorleveloptions()

Sets the errorlevel options.

## setformats (formats, usetemplates)

Sets the format options using the given format dictionary.

#### **Parameters** formats (Dictionary) – The dictionary keys should be:

- Single strings (or 1-tuples) containing an input format (if not *usetemplates*)
- Tuples containing an input format and template format (if *usetemplates*)
- Formats can be *None* to indicate what to do with standard input

The dictionary *values* should be tuples of outputformat (string) and processor method.

```
setmanpageoption()
     creates a manpage option that allows the optionparser to generate a manpage
setprogressoptions()
     Sets the progress options.
splitext (pathname)
     Splits pathname into name and ext, and removes the extsep.
         Parameters pathname (string) – A file path
         Returns root, ext
         Return type tuple
splitinputext (inputpath)
     Splits an inputpath into name and extension.
splittemplateext (templatepath)
     Splits a templatepath into name and extension.
templateexists (options, templatepath)
     Returns whether the given template exists...
warning (msg, options=None, exc_info=None)
     Print a warning message incorporating 'msg' to stderr and exit.
```

#### posegment

Segment Gettext PO, XLIFF and TMX localization files at the sentence level.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/posegment.html for examples and usage instructions.

```
translate. tools.posegment. \textbf{segmentfile} (inputfile, outputfile, templatefile, source-language='en', targetlanguage=None, stripspaces=True, onlyaligned=False) \\ reads in inputfile, segments it then, writes to outputfile
```

## poswap

Builds a new translation file with the target of the input language as source language.

**Note:** Ensure that the two po files correspond 100% to the same pot file before using this.

To translate Kurdish (ku) through French:

```
poswap -i fr/ -t ku -o fr-ku
```

To convert the fr-ku files back to en-ku:

```
poswap --reverse -i fr/ -t fr-ku -o en-ku
```

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/poswap.html for examples and usage instructions.

```
translate.tools.poswap.convertpo (inputpofile, outputpotfile, template, reverse=False) reads in inputpofile, removes the header, writes to outputpotfile.
```

4.1. API 693

```
translate.tools.poswap.swapdir(store)
```

Swap the source and target of each unit.

## poterminology

Create a terminology file by reading a set of .po or .pot files to produce a pootle-terminology.pot.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/poterminology.html for examples and usage instructions.

```
class translate.tools.poterminology.TerminologyOptionParser (formats, usetem-
plates=False,
allowmissingtem-
plate=False, de-
scription=None)

a specialized Option Parser for the terminology tool...

add_option(Option)
```

```
add_option(opt_str, ..., kwarg=val, ...)
check_values (values : Values, args : [string])
    -> (values : Values, args : [string])
```

Check that the supplied option values and leftover arguments are valid. Returns the option values and leftover arguments (possibly adjusted, possibly completely new – whatever you like). Default implementation just returns the passed-in values; subclasses may override as desired.

## checkoutputsubdir(options, subdir)

Checks to see if subdir under options.output needs to be created, creates if neccessary.

#### define\_option(option)

Defines the given option, replacing an existing one of the same short name if neccessary...

#### destroy()

Declare that you are done with this OptionParser. This cleans up reference cycles so the OptionParser (and all objects referenced by it) can be garbage-collected promptly. After calling destroy(), the OptionParser is unusable.

## disable\_interspersed\_args()

Set parsing to stop on the first non-option. Use this if you have a command processor which runs another command that has options of its own and you want to make sure these options don't get confused.

## enable\_interspersed\_args()

Set parsing to not stop on the first non-option, allowing interspersing switches with command arguments. This is the default behavior. See also disable\_interspersed\_args() and the class documentation description of the attribute allow\_interspersed\_args.

```
error (msg : string)
```

Print a usage message incorporating 'msg' to stderr and exit. If you override this in a subclass, it should not return – it should either exit or raise an exception.

## $\begin{tabular}{ll} \textbf{finalizetempoutputfile} (options, output file, full output path) \\ \end{tabular}$

Write the temp outputfile to its final destination.

```
format_manpage()
```

returns a formatted manpage

## getformathelp(formats)

Make a nice help string for describing formats...

## getfullinputpath (options, inputpath) Gets the absolute path to an input file. getfulloutputpath (options, outputpath) Gets the absolute path to an output file. getfulltemplatepath (options, templatepath) Gets the absolute path to a template file. getoutputname (options, inputname, outputformat) Gets an output filename based on the input filename. getoutputoptions (options, inputpath, templatepath) Works out which output format and processor method to use... getpassthroughoptions (options) Get the options required to pass to the filtermethod... gettemplatename (options, inputname) Gets an output filename based on the input filename. getusageman (option) returns the usage string for the given option getusagestring(option) returns the usage string for the given option isexcluded (options, inputpath) Checks if this path has been excluded. isrecursive (fileoption, filepurpose='input') Checks if fileoption is a recursive file. isvalidinputname (inputname) Checks if this is a valid input filename. **mkdir** (parent, subdir) Makes a subdirectory (recursively if neccessary). openinputfile (options, fullinputpath) Opens the input file. openoutputfile (options, fulloutputpath) Opens the output file. opentemplatefile (options, fulltemplatepath) Opens the template file (if required). opentempoutputfile (options, fulloutputpath) Opens a temporary output file. outputterminology (options) saves the generated terminology glossary parse\_args (args=None, values=None) parses the command line options, handling implicit input/output args print\_help(file : file = stdout) Print an extended help message, listing all options and any help text provided with them, to 'file' (default stdout). print manpage (file=None)

4.1. API 695

outputs a manpage for the program using the help information

## print\_usage (file : file = stdout)

Print the usage message for the current program (self.usage) to 'file' (default stdout). Any occurrence of the string "%prog" in self.usage is replaced with the name of the current program (basename of sys.argv[0]). Does nothing if self.usage is empty or not defined.

#### print\_version (file : file = stdout)

Print the version message for this program (self.version) to 'file' (default stdout). As with print\_usage(), any occurrence of "%prog" in self.version is replaced by the current program's name. Does nothing if self.version is empty or undefined.

#### processfile (fileprocessor, options, fullinputpath)

process an individual file

#### recurseinputfilelist (options)

Use a list of files, and find a common base directory for them.

#### recurseinputfiles (options)

Recurse through directories and return files to be processed.

#### recursiveprocess (options)

recurse through directories and process files

#### run()

parses the arguments, and runs recursive process with the resulting options

#### set\_usage (usage=None)

sets the usage string - if usage not given, uses getusagestring for each option

#### seterrorleveloptions()

Sets the errorlevel options.

#### setformats (formats, usetemplates)

Sets the format options using the given format dictionary.

**Parameters formats** (Dictionary) – The dictionary keys should be:

- Single strings (or 1-tuples) containing an input format (if not *usetemplates*)
- Tuples containing an input format and template format (if *usetemplates*)
- Formats can be *None* to indicate what to do with standard input

The dictionary *values* should be tuples of outputformat (string) and processor method.

## setmanpageoption()

creates a manpage option that allows the optionparser to generate a manpage

#### setprogressoptions()

Sets the progress options.

#### splitext (pathname)

Splits *pathname* into name and ext, and removes the extsep.

## Parameters pathname (string) - A file path

Returns root, ext

Return type tuple

#### splitinputext (inputpath)

Splits an *inputpath* into name and extension.

## splittemplateext(templatepath)

Splits a *templatepath* into name and extension.

```
templateexists (options, templatepath)
     Returns whether the given template exists...
warning (msg, options=None, exc_info=None)
     Print a warning message incorporating 'msg' to stderr and exit.
```

## pretranslate

Fill localization files with suggested translations based on translation memory and existing translations.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/pretranslate.html for examples and

```
usage instructions.
translate.tools.pretranslate.match_fuzzy(input_unit, matchers)
     Return a fuzzy match from a queue of matchers.
translate.tools.pretranslate.match_source(input_unit, template_store)
     Returns a matching unit from a template. matching based on unit id
translate.tools.pretranslate.match template id(input unit, template store)
     Returns a matching unit from a template. matching based on unit id
translate.tools.pretranslate.match_template_location(input_unit, template_store)
     Returns a matching unit from a template. matching based on locations
translate.tools.pretranslate.memory (tmfiles,
                                                         max \ candidates=1,
                                                                               min \ similarity=75,
                                              max length=1000)
     Returns the TM store to use. Only initialises on first call.
translate.tools.pretranslate.pretranslate file (input file, output file, template file,
                                                             tm=None, min_similarity=75, fuzzy-
                                                             matching=True)
     Pretranslate any factory supported file with old translations and translation memory.
translate.tools.pretranslate.pretranslate_store(input_store, template_store, tm=None,
                                                              min_similarity=75,
                                                                                      fuzzymatch-
                                                              ing=True)
     Do the actual pretranslation of a whole store.
```

```
translate.tools.pretranslate.pretranslate_unit(input_unit,
                                                                     template_store,
                                                                                    match-
                                                                         mark_reused=False,
                                                         ers=None,
                                                         merge on='id')
```

Pretranslate a unit or return unchanged if no translation was found.

#### **Parameters**

- input\_unit Unit that will be pretranslated.
- template\_store Fill input unit with units matching in this store.
- matchers List of fuzzy matcher objects.
- mark reused Whether to mark old translations as reused or not.
- merge\_on Where will the merge matching happen on.

## pydiff

diff tool like GNU diff, but lets you have special options that are useful in dealing with PO files

4.1. API 697

```
class translate.tools.pydiff.DirDiffer (fromdir, todir, options)
     generates diffs between directories
     isexcluded(difffile)
          checks if the given filename has been excluded from the diff
     writediff(outfile)
          writes the actual diff to the given file
class translate.tools.pydiff.FileDiffer (fromfile, tofile, options)
     generates diffs between files
     get_from_lines(group)
          returns the lines referred to by group, from the fromfile
     get_to_lines(group)
          returns the lines referred to by group, from the tofile
     unified_diff(group)
          takes the group of opcodes and generates a unified diff line by line
     writediff(outfile)
          writes the actual diff to the given file
translate.tools.pydiff.main()
     main program for pydiff
pypo2phppo
Convert Python format .po files to PHP format .po files.
translate.tools.pypo2phppo.convertpy2php(inputfile, outputfile, template=None)
     Converts from Python .po to PHP .po
          Parameters
```

- inputfile file handle of the source
- outputfile file handle to write to
- template unused

translate.tools.pypo2phppo.main(argv=None)
Converts from Python.po to PHP.po

```
translate.convert.po2wordfast, 258
                                           translate.convert.po2xliff, 262
translate.convert, 231
                                           translate.convert.po2yaml, 262
translate.convert.accesskey, 232
                                           translate.convert.pot2po,263
translate.convert.convert, 233
                                           translate.convert.prop2mozfunny, 263
translate.convert.csv2po, 240
                                           translate.convert.prop2po, 264
translate.convert.csv2tbx, 241
                                           translate.convert.rc2po, 264
translate.convert.dtd2po, 241
                                           translate.convert.resx2po, 265
translate.convert.factory, 242
                                           translate.convert.sub2po, 265
translate.convert.html2po,242
                                           translate.convert.symb2po, 266
translate.convert.ical2po, 242
                                           translate.convert.tiki2po, 266
translate.convert.ini2po, 243
                                           translate.convert.ts2po, 266
translate.convert.json2po, 243
                                           translate.convert.txt2po, 266
translate.convert.moz2po, 244
                                           translate.convert.web2py2po,267
translate.convert.mozfunny2prop, 244
                                           translate.convert.xliff2odf,267
translate.convert.mozlang2po, 244
                                           translate.convert.xliff2oo, 267
translate.convert.odf2xliff, 245
                                           translate.convert.xliff2po, 268
translate.convert.oo2po, 245
                                           translate.convert.yaml2po, 268
translate.convert.oo2xliff,245
                                           translate.filters, 231
translate.convert.php2po, 245
                                           translate.filters.autocorrect, 268
translate.convert.po2csv, 246
                                           translate.filters.checks, 269
translate.convert.po2dtd, 246
                                           translate.filters.decoration, 347
translate.convert.po2html, 246
                                           translate.filters.helpers, 348
translate.convert.po2ical, 247
                                           translate.filters.pofilter, 348
translate.convert.po2ini,247
                                           translate.filters.prefilters, 351
translate.convert.po2json, 248
                                           translate.filters.spelling, 352
translate.convert.po2moz, 249
                                           translate.lang, 352
translate.convert.po2mozlang, 248
                                           translate.lang.af, 352
translate.convert.po2oo, 252
                                           translate.lang.am, 353
translate.convert.po2php, 252
                                           translate.lang.ar, 354
translate.convert.po2prop, 252
                                           translate.lang.bn, 355
translate.convert.po2rc, 253
                                           translate.lang.code or, 356
translate.convert.po2resx, 253
                                           translate.lang.common, 356
translate.convert.po2sub, 253
                                           translate.lang.data, 360
translate.convert.po2symb, 253
                                           translate.lang.de, 363
translate.convert.po2tiki, 253
                                           translate.lang.el, 363
translate.convert.po2tmx, 254
                                           translate.lang.es, 364
translate.convert.po2ts, 258
                                           translate.lang.fa, 365
translate.convert.po2txt, 258
                                           translate.lang.factory, 365
translate.convert.po2web2py, 258
                                           translate.lang.fi, 366
```

```
translate.lang.fr, 367
                                          translate.storage._factory_classes,431
translate.lang.gu, 368
                                          translate.storage.base, 408
translate.lang.he, 368
                                          translate.storage.benchmark, 413
translate.lang.hi,369
                                          translate.storage.bundleprojstore, 413
translate.lang.hy, 370
                                          translate.storage.catkeys, 414
translate.lang.identify, 371
                                          translate.storage.csvl10n,420
translate.lang.ja,371
                                          translate.storage.directory, 425
translate.lang.km, 372
                                          translate.storage.dtd, 425
translate.lang.kn, 372
                                          translate.storage.factory, 431
translate.lang.ko, 373
                                          translate.storage.html, 432
translate.lang.ml, 374
                                          translate.storage.ical, 441
translate.lang.mr, 375
                                          translate.storage.ini,446
translate.lang.ne, 375
                                          translate.storage.jsonl10n,451
translate.lang.ngram, 376
                                          translate.storage.lisa,471
translate.lang.pa, 376
                                          translate.storage.mo, 476
translate.lang.poedit, 377
                                          translate.storage.mozilla_lang,482
translate.lang.si, 378
                                          translate.storage.odf_io,487
translate.lang.st, 379
                                          translate.storage.odf shared, 487
translate.lang.sv, 379
                                          translate.storage.omegat, 487
translate.lang.ta, 380
                                          translate.storage.oo, 494
translate.lang.te, 381
                                          translate.storage.php, 546
translate.lang.team, 381
                                          translate.storage.placeables, 497
translate.lang.th, 382
                                          translate.storage.placeables.base, 497
translate.lang.ug.383
                                          translate.storage.placeables.general,
translate.lang.ur, 383
                                                  512
translate.lang.vi, 384
                                          translate.storage.placeables.interfaces,
translate.lang.zh, 385
                                                  517
translate.misc, 386
                                          translate.storage.placeables.lisa, 525
translate.misc.dictutils, 386
                                          translate.storage.placeables.parse, 525
translate.misc.file_discovery, 387
                                          translate.storage.placeables.strelem,
translate.misc.lru, 387
translate.misc.multistring, 387
                                          translate.storage.placeables.terminology,
translate.misc.optrecurse, 390
translate.misc.ourdom, 393
                                          translate.storage.placeables.xliff,530
translate.misc.progressbar, 394
                                          translate.storage.po, 559
translate.misc.quote, 394
                                          translate.storage.pocommon, 552
translate.misc.wsgi, 395
                                          translate.storage.poheader, 558
translate.misc.wStringIO, 395
                                          translate.storage.poparser, 559
translate.misc.xml helpers, 396
                                          translate.storage.poxliff,560
translate.search, 397
                                          translate.storage.project, 567
translate.search.indexing, 397
                                          translate.storage.projstore, 568
translate.search.indexing.CommonIndexer, translate.storage.properties, 569
                                          translate.storage.pypo,588
translate.search.indexing.PyLuceneIndexetranslate.storage.qm, 595
                                          translate.storage.qph,600
translate.search.indexing.XapianIndexer, translate.storage.rc, 606
                                          translate.storage.statistics,610
translate.search.lshtein,405
                                          translate.storage.statsdb,611
translate.search.match, 405
                                          translate.storage.subtitles,612
translate.search.segment, 407
                                          translate.storage.symbian, 624
                                          translate.storage.tbx, 624
translate.search.terminology, 407
translate.services, 407
                                          translate.storage.tiki,629
                                          translate.storage.tmdb,634
translate.services.tmserver, 407
translate.storage, 408
                                          translate.storage.tmx, 634
```

```
translate.storage.trados,639
translate.storage.ts,651
translate.storage.ts2,645
translate.storage.txt,651
translate.storage.utx,656
translate.storage.versioncontrol,661
translate.storage.versioncontrol.bzr,
translate.storage.versioncontrol.cvs,
      663
translate.storage.versioncontrol.darcs,
translate.storage.versioncontrol.git,
translate.storage.versioncontrol.hg,664
translate.storage.versioncontrol.svn,
      665
translate.storage.wordfast,665
translate.storage.workflow,672
translate.storage.xliff,672
translate.storage.xml_extract,678
translate.storage.xml_extract.extract,
      678
translate.storage.xml extract.generate,
      679
translate.storage.xml_extract.misc,680
translate.storage.xml_extract.unit_tree,
translate.storage.xml_extract.xpath_breadcrumb,
translate.storage.xml_name,681
translate.storage.zip,681
translate.tools, 682
translate.tools.build_tmdb,682
translate.tools.phppo2pvpo,682
translate.tools.poclean, 682
translate.tools.pocompile, 682
translate.tools.poconflicts, 683
translate.tools.pocount,686
translate.tools.podebug, 686
translate.tools.pogrep, 686
translate.tools.pomerge, 690
translate.tools.porestructure, 690
translate.tools.posegment, 693
translate.tools.poswap, 693
translate.tools.poterminology, 694
translate.tools.pretranslate, 697
translate.tools.pydiff,697
translate.tools.pypo2phppo,698
```

A	accesskeysuffixes (in module trans-
accelerators() (trans-	late.storage.dtd), 426
late.filters.checks.CCLicenseChecker method),	accesskeysuffixes (in module trans-
269	late.storage.properties), 574
accelerators() (trans-	acronyms () (translate.filters.checks.CCLicenseChecker
late.filters.checks.DrupalChecker method),	method), 269
275	acronyms () (translate.filters.checks.DrupalChecker
accelerators() (trans-	method), 275
late.filters.checks.GnomeChecker method), 281	acronyms () (translate.filters.checks.GnomeChecker method), 281
accelerators()(translate.filters.checks.IOSChecker	acronyms() (translate.filters.checks.IOSChecker
method), 286	method), 287
accelerators() (trans-	acronyms() (translate.filters.checks.KdeChecker
late.filters.checks.KdeChecker method),	method), 292
292	acronyms() (translate.filters.checks.L20nChecker
accelerators() (trans-	method), 298
late.filters.checks.L20nChecker method),	acronyms () (translate.filters.checks.LibreOfficeChecker
298	method), 304
accelerators() (trans-	acronyms () (translate.filters.checks.MinimalChecker
$late. filters. checks. Libre Of fice Checker\ method),$	method), 309
304	acronyms () (translate.filters.checks.MozillaChecker
accelerators() (trans-	method), 315
late.filters.checks.MinimalChecker method),	acronyms () (translate.filters.checks.OpenOfficeChecker
309	method), 321
accelerators() (trans-	acronyms () (translate.filters.checks.ReducedChecker method), 327
late.filters.checks.MozillaChecker method),	acronyms () (translate.filters.checks.StandardChecker
315	method), 333
accelerators() (trans-	acronyms() (translate.filters.checks.TermChecker
late.filters.checks.OpenOfficeChecker method),	method), 340
321	add() (translate.storage.versioncontrol.bzr.bzr method),
accelerators() (trans-	663
late.filters.checks.ReducedChecker method), 327	add() (translate.storage.versioncontrol.cvs.cvs
accelerators() (trans-	method), 663
late.filters.checks.StandardChecker method),	add() (translate.storage.versioncontrol.darcs.darcs
332	method), 664
accelerators() (trans-	add() (translate.storage.versioncontrol.GenericRevisionControlSystem
late.filters.checks.TermChecker method),	method), 662
339	add() (translate.storage.versioncontrol.git.git method),
	664

add()	(translate.storage.versioncontrol.hg.hg n	rethod),	me	ethod), 237	
	664		add_optic		(trans-
add()	(translate.storage.versioncontrol. method), 665	.svn.svn		e.convert.po2moz.MozConvertOptionthod), 249	onParser
add_dı	uplicates_option()	(trans-	add_optic		(trans-
	late.convert.convert.ArchiveConvertOpt method), 233		· lat	e.convert.po2tmx.TmxOptionParser athod), 254	
add_dı	uplicates_option()	(trans-	add_optic	on ()	(trans-
	late.convert.convert.ConvertOptionPars method), 237	er		e.convert.po2wordfast.WfOptionPa ethod), 259	rser
add_dı	uplicates_option()	(trans-	add_optic		(trans-
	late.convert.po2moz.MozConvertOption method), 249	Parser	34		method),
add_dı	uplicates_option()	(trans-	add_optic	on ()	(trans-
	late.convert.po2tmx.TmxOptionParser method), 254			e.misc.optrecurse.RecursiveOption. ethod), 390	Parser
add_dı	uplicates_option()	(trans-	add_optic		(trans-
	late.convert.po2wordfast.WfOptionParsmethod), 259	er		e.tools.poconflicts.ConflictOptionPo ethod), 683	arser
add_f	uzzy_option()		add_optic		(trans-
	late.convert.convert.ArchiveConvertOpt method), 233	ionParser	· lat	e.tools.pogrep.GrepOptionParser 7	method),
add_f	uzzy_option()	(trans-	add_optic		(trans-
	late.convert.convert.ConvertOptionPars method), 237	er		e.tools.porestructure.SplitOptionPa ethod), 690	rser
add_f	uzzy_option()     late.convert.po2moz.MozConvertOption     method), 249	(trans- Parser		on () e.tools.poterminology.Terminology ethod), 694	(trans- OptionParser
add_f	<pre>uzzy_option()   late.convert.po2tmx.TmxOptionParser   method), 254</pre>	(trans-	lat	ve_untranslated_option() e.convert.convert.ArchiveConvertO ethod), 233	
add_fi	uzzy_option()	(trans-	add_remov	<pre>ve_untranslated_option()</pre>	(trans-
	late.convert.po2wordfast.WfOptionParsemethod), 259	er		e.convert.convert.ConvertOptionPa ethod), 237	rser
add_mu	ultifile_option()     late.convert.convert.ArchiveConvertOpt     method), 233		· lat	ve_untranslated_option() e.convert.po2moz.MozConvertOption ethod), 249	(trans- onParser
	ultifile_option()  late.convert.convert.ConvertOptionPars  method), 237		lat me	ve_untranslated_option() e.convert.po2tmx.TmxOptionParser ethod), 254	•
add_mı	ultifile_option()	(trans-		ve_untranslated_option()	
	late.convert.po2moz.MozConvertOption method), 249	Parser		e.convert.po2wordfast.WfOptionPa ethod), 259	
add_mı	ultifile_option()	(trans-	add_sour		ct.Project
	late.convert.po2tmx.TmxOptionParser			ethod), 567	(,
2 d d ===	method), 254	(trans		ce_convert()	(trans-
auu_ini	ultifile_option()     late.convert.po2wordfast.WfOptionPars	(trans-		e.storage.project.Project method),	(trans-
	method), 259	c i		e.storage.csvl10n.csvunit method),	,
add or	otion()	(trans-		shold_option()	(trans-
	late.convert.convert.ArchiveConvertOpt method), 233	`	· lat	e.convert.convert.ArchiveConvertO ethod), 233	`
add or	otion()	(trans-		shold_option()	(trans-
1	late.convert.convert.ConvertOptionPars			e.convert.convert.ConvertOptionPa	,

method), 237	late.storage.php.phpfile method), 547
add_threshold_option() (trans-	add_unit_to_index() (trans-
late.convert.po2moz.MozConvertOptionParser	late.storage.pocommon.pofile method), 552
method), 249	add_unit_to_index() (trans-
add_threshold_option() (trans-	late.storage.poxliff.PoXliffFile method), 560
late.convert.po2tmx.TmxOptionParser	add_unit_to_index() (trans-
method), 254	late.storage.properties.javafile method),
add_threshold_option() (trans-	575
late.convert.po2wordfast.WfOptionParser	add_unit_to_index() (trans-
<pre>method), 259 add_unit_to_index() (trans-</pre>	late.storage.properties.javautf8file method), 577
<pre>add_unit_to_index() (trans- late.storage.base.TranslationStore method),</pre>	add_unit_to_index() (trans-
408	late.storage.properties.joomlafile method),
add_unit_to_index() (trans-	578
late.storage.catkeys.CatkeysFile method),	add_unit_to_index() (trans-
415	late.storage.properties.propfile method),
add_unit_to_index() (trans-	580
late.storage.csvl10n.csvfile method), 420	add_unit_to_index() (trans-
add_unit_to_index() (translate.storage.dtd.dtdfile	late.storage.properties.stringsfile method),
method), 426	585
add_unit_to_index() (trans-	add_unit_to_index() (trans-
late.storage.html.htmlfile method), 435	late.storage.properties.stringsutf8file method),
add_unit_to_index() (trans-	586
late.storage.html.POHTMLParser method),	add_unit_to_index() (trans-
432	late.storage.pypo.pofile method), 588
<pre>add_unit_to_index() (trans- late.storage.ical.icalfile method), 441</pre>	<pre>add_unit_to_index() (translate.storage.qm.qmfile</pre>
<pre>add_unit_to_index() (translate.storage.ini.inifile</pre>	add_unit_to_index() (trans-
method), 446	late.storage.qph.QphFile method), 600
add_unit_to_index() (trans-	<pre>add_unit_to_index() (translate.storage.rc.rcfile</pre>
late.storage.jsonl10n.I18NextFile method),	method), 606
452	add_unit_to_index() (trans-
add_unit_to_index() (trans-	late. storage. subtitles. Adv SubStation Alpha File
late.storage.jsonl10n.JsonFile method), 456	method), 612
add_unit_to_index() (trans-	add_unit_to_index() (trans-
late.storage.jsonl10n.JsonNestedFile method), 458	late.storage.subtitles.MicroDVDFile method), 614
add_unit_to_index() (trans-	add_unit_to_index() (trans-
late. storage. js on 110 n. Web Extension Js on File	$late. storage. subtitles. SubRipFile \\ method),$
method), 466	616
add_unit_to_index() (trans-	add_unit_to_index() (trans-
late.storage.lisa.LISAfile method), 471	late. storage. subtitles. SubStation Alpha File
add_unit_to_index() (translate.storage.mo.mofile	method), 617
method), 476	add_unit_to_index() (trans-
add_unit_to_index() (trans-	late.storage.subtitles.SubtitleFile method),
late.storage.mozilla_lang.LangStore method),	619
482	add_unit_to_index() (translate.storage.tbx.tbxfile
add_unit_to_index() (trans-	method), 624
late.storage.omegat.OmegaTFile method), 488	add_unit_to_index() (trans-
add_unit_to_index() (trans-	<pre>late.storage.tiki.TikiStore method), 630 add_unit_to_index() (trans-</pre>
late.storage.omegat.OmegaTFileTab method),	late.storage.tmx.tmxfile method), 634
489	add_unit_to_index() (trans-
add_unit_to_index() (trans-	late.storage.trados.TradosTxtTmFile method),
	· · · · · · · · · · · · · · · · · · ·

643	591
add_unit_to_index() (translate.storage.ts2.tsfile method), 645	adderror() (translate.storage.qm.qmunit method), 597
add_unit_to_index() (trans- late.storage.txt.TxtFile method), 651	adderror() (translate.storage.qph.QphUnit method), 602
add_unit_to_index() (trans- late.storage.utx.UtxFile method), 656	adderror() (translate.storage.rc.rcunit method), 607 adderror() (translate.storage.subtitles.SubtitleUnit
add_unit_to_index() (trans-	method), 621
late.storage.wordfast.WordfastTMFile method), 667	adderror() (translate.storage.tbx.tbxunit method), 626
<pre>add_unit_to_index() (trans- late.storage.xliff.xlifffile method), 672</pre>	adderror() (translate.storage.tiki.TikiUnit method), 631
addaltrans() (translate.storage.poxliff.PoXliffUnit method), 563	adderror() (translate.storage.tmx.tmxunit method), 636
addalttrans() (translate.storage.xliff.xliffunit method), 675	adderror() (translate.storage.trados.TradosUnit method), 640
adderror() (translate.storage.base.TranslationUnit	adderror() (translate.storage.ts2.tsunit method), 647
method), 410	adderror() (translate.storage.txt.TxtUnit method),
adderror() (translate.storage.catkeys.CatkeysUnit	653
method), 416 adderror() (translate.storage.csvl10n.csvunit	adderror() (translate.storage.utx.UtxUnit method), 658
method), 422	adderror() (translate.storage.wordfast.WordfastUnit
adderror() (translate.storage.dtd.dtdunit method),	method), 669
428 adderror() (translate.storage.html.htmlunit method),	adderror() (translate.storage.xliff.xliffunit method), 675
437 adderror() (translate.storage.ical.icalunit method),	addheader() (translate.storage.lisa.LISAfile method), 471
443	addheader() (translate.storage.poxliff.PoXliffFile
adderror() (translate.storage.ini.iniunit method), 448	method), 560
adderror() (translate.storage.ini.iniunit method), 448 adderror() (translate.storage.jsonl10n.118NextUnit method), 453	method), 560 addheader() (translate.storage.qph.QphFile method), 600
adderror() (translate.storage.ini.iniunit method), 448 adderror() (translate.storage.jsonl10n.118NextUnit	method), 560 addheader() (translate.storage.qph.QphFile method), 600
adderror() (translate.storage.ini.iniunit method), 448 adderror() (translate.storage.jsonl10n.118NextUnit method), 453 adderror() (translate.storage.jsonl10n.JsonNestedUnit method), 460 adderror() (translate.storage.jsonl10n.JsonUnit	method), 560 addheader() (translate.storage.qph.QphFile method), 600 addheader() (translate.storage.tbx.tbxfile method), 624 addheader() (translate.storage.tmx.tmxfile method),
adderror() (translate.storage.ini.iniunit method), 448 adderror() (translate.storage.jsonl10n.I18NextUnit method), 453 adderror() (translate.storage.jsonl10n.JsonNestedUnit method), 460	method), 560 addheader() (translate.storage.qph.QphFile method), 600 addheader() (translate.storage.tbx.tbxfile method), 624 addheader() (translate.storage.tmx.tmxfile method), 634
adderror() (translate.storage.ini.iniunit method), 448 adderror() (translate.storage.jsonl10n.118NextUnit	method), 560 addheader() (translate.storage.qph.QphFile method), 600 addheader() (translate.storage.tbx.tbxfile method), 624 addheader() (translate.storage.tmx.tmxfile method), 634 southireader() (translate.storage.ts2.tsfile method), 645 addheader() (translate.storage.xliff.xlifffile method),
adderror() (translate.storage.ini.iniunit method), 448 adderror() (translate.storage.jsonl10n.118NextUnit	method), 560 addheader() (translate.storage.qph.QphFile method), 600 addheader() (translate.storage.tbx.tbxfile method), 624 addheader() (translate.storage.tmx.tmxfile method), 634 combiniteader() (translate.storage.ts2.tsfile method), 645
adderror() (translate.storage.ini.iniunit method), 448 adderror() (translate.storage.jsonl10n.I18NextUnit	method), 560 addheader() (translate.storage.qph.QphFile method), 600 addheader() (translate.storage.tbx.tbxfile method), 624 addheader() (translate.storage.tmx.tmxfile method), 634 condditieader() (translate.storage.ts2.tsfile method), 645 addheader() (translate.storage.xliff.xlifffile method), 672 addline() (translate.storage.oo.oofile method), 495 addline() (translate.storage.oo.oounit method), 496
adderror() (translate.storage.ini.iniunit method), 448 adderror() (translate.storage.jsonl10n.I18NextUnit	method), 560 addheader() (translate.storage.qph.QphFile method), 600 addheader() (translate.storage.tbx.tbxfile method), 624 addheader() (translate.storage.tmx.tmxfile method), 634 soutbhireader() (translate.storage.ts2.tsfile method), 645 addheader() (translate.storage.xliff.xlifffile method), 672 addline() (translate.storage.oo.oofile method), 495 addline() (translate.storage.oo.oofile method), 496 addlocation() (translate.storage.base.TranslationUnit method),
adderror() (translate.storage.ini.iniunit method), 448 adderror() (translate.storage.jsonl10n.I18NextUnit	method), 560 addheader() (translate.storage.qph.QphFile method), 600 addheader() (translate.storage.tbx.tbxfile method), 624 addheader() (translate.storage.tmx.tmxfile method), 634 southineader() (translate.storage.ts2.tsfile method), 645 addheader() (translate.storage.xliff.xlifffile method), 672 addline() (translate.storage.oo.oofile method), 495 addline() (translate.storage.oo.oomit method), 496 addlocation() (translate.storage.base.TranslationUnit method), 410 addlocation() (trans-
adderror() (translate.storage.ini.iniunit method), 448 adderror() (translate.storage.jsonl10n.II8NextUnit	method), 560 addheader() (translate.storage.qph.QphFile method), 600 addheader() (translate.storage.tbx.tbxfile method), 624 addheader() (translate.storage.tmx.tmxfile method), 634 southiteader() (translate.storage.ts2.tsfile method), 645 addheader() (translate.storage.xliff.xlifffile method), 672 addline() (translate.storage.oo.oofile method), 495 addline() (translate.storage.oo.oomit method), 496 addlocation() (translate.storage.base.TranslationUnit method), 410 addlocation() (translate.storage.catkeys.CatkeysUnit method),
adderror() (translate.storage.ini.iniunit method), 448 adderror() (translate.storage.jsonl10n.I18NextUnit	method), 560 addheader() (translate.storage.qph.QphFile method), 600 addheader() (translate.storage.tbx.tbxfile method), 624 addheader() (translate.storage.tmx.tmxfile method), 634 southineader() (translate.storage.ts2.tsfile method), 645 addheader() (translate.storage.xliff.xlifffile method), 672 addline() (translate.storage.oo.oofile method), 495 addline() (translate.storage.oo.oomit method), 496 addlocation() (translate.storage.base.TranslationUnit method), 410 addlocation() (trans-
adderror() (translate.storage.ini.iniunit method), 448 adderror() (translate.storage.jsonl10n.I18NextUnit	method), 560 addheader() (translate.storage.qph.QphFile method), 600 addheader() (translate.storage.tbx.tbxfile method), 624 addheader() (translate.storage.tmx.tmxfile method), 634 southwiteader() (translate.storage.ts2.tsfile method), 645 addheader() (translate.storage.xliff.xlifffile method), 672 addline() (translate.storage.oo.oofile method), 495 addline() (translate.storage.oo.oomit method), 496 addlocation() (translate.storage.base.TranslationUnit method), 410 addlocation() (translate.storage.catkeys.CatkeysUnit method), 417 addlocation() (translate.storage.csvl10n.csvunit method), 422
adderror() (translate.storage.ini.iniunit method), 448 adderror() (translate.storage.jsonl10n.II8NextUnit	method), 560 addheader() (translate.storage.qph.QphFile method), 600 addheader() (translate.storage.tbx.tbxfile method), 624 addheader() (translate.storage.tmx.tmxfile method), 634 condditionader() (translate.storage.ts2.tsfile method), 645 addheader() (translate.storage.xliff.xlifffile method), 672 addline() (translate.storage.oo.oofile method), 495 addline() (translate.storage.oo.oounit method), 496 addlocation() (translate.storage.base.TranslationUnit method), 410 addlocation() (translate.storage.catkeys.CatkeysUnit method), 417 addlocation() (translate.storage.csvl10n.csvunit
adderror() (translate.storage.ini.iniunit method), 448 adderror() (translate.storage.jsonl10n.II8NextUnit	method), 560 addheader() (translate.storage.qph.QphFile method), 600 addheader() (translate.storage.tbx.tbxfile method), 624 addheader() (translate.storage.tmx.tmxfile method), 634 southinteader() (translate.storage.ts2.tsfile method), 645 addheader() (translate.storage.xliff.xlifffile method), 672 addline() (translate.storage.oo.oofile method), 495 addline() (translate.storage.oo.oounit method), 496 addlocation() (translate.storage.base.TranslationUnit method), 410 addlocation() (translate.storage.csvl10n.csvunit method), 422 addlocation() (translate.storage.csvl10n.csvunit method), 422 addlocation() (translate.storage.dtd.dtdunit
adderror() (translate.storage.ini.iniunit method), 448 adderror() (translate.storage.jsonl10n.II8NextUnit	method), 560 addheader () (translate.storage.qph.QphFile method), 600 addheader () (translate.storage.tbx.tbxfile method), 624 addheader () (translate.storage.tmx.tmxfile method), 634 southiteader () (translate.storage.ts2.tsfile method), 645 addheader () (translate.storage.xliff.xlifffile method), 672 addline () (translate.storage.oo.oofile method), 495 addline () (translate.storage.oo.oomit method), 496 addlocation () (translate.storage.base.TranslationUnit method), 410 addlocation () (translate.storage.csvl10n.csvunit method), 422 addlocation () (translate.storage.csvl10n.csvunit method), 428 addlocation () (translate.storage.dtd.dtdunit method), 428 addlocation () (translate.storage.html.htmlunit method), 437
adderror() (translate.storage.ini.iniunit method), 448 adderror() (translate.storage.jsonl10n.II8NextUnit	method), 560 addheader() (translate.storage.qph.QphFile method), 600 addheader() (translate.storage.tbx.tbxfile method), 624 addheader() (translate.storage.tmx.tmxfile method), 634 southiteader() (translate.storage.ts2.tsfile method), 645 addheader() (translate.storage.xliff.xlifffile method), 672 addline() (translate.storage.oo.oofile method), 495 addline() (translate.storage.oo.oomit method), 496 addlocation() (translate.storage.base.TranslationUnit method), 410 addlocation() (translate.storage.csvl10n.csvunit method), 422 addlocation() (translate.storage.csvl10n.csvunit method), 422 addlocation() (translate.storage.dtd.dtdunit method), 428 addlocation() (translate.storage.html.htmlunit

addlocation()(tran	nslate.storage.ini.iniunit meti	hod),	method), 653	
448			addlocation()	(translate.storage.utx.UtxUnit
addlocation()	(ti	rans-	method), 658	
late.storage.jsc	onl10n.I18NextUnit meth	hod),	addlocation()	(trans-
453			late.storage.w	ordfast. Wordfast Unit method),
addlocation()	(ti	rans-	669	
late.storage.jse 460	onl10n.JsonNestedUnit meth	hod),	addlocation()  method), 675	(translate.storage.xliff.xliffunit
addlocation() (tra	nslate.storage.jsonl10n.Json	ıUnit	addlocations()	(trans-
method), 463	0 0			use.TranslationUnit method),
addlocation()	(ti	rans-	410	,,
	onl10n.WebExtensionJsonÙr	ıit	addlocations()	(trans-
method), 468				atkeys.CatkeysUnit method),
addlocation()	(translate.storage.lisa.LISA	Aunit	417	,,
method), 473	(			translate.storage.csvl10n.csvunit
addlocation()	(translate.storage.mo.m	ounit	method), 422	
method), 479	(ir ansatare isra ragemia in		addlocations()	(translate.storage.dtd.dtdunit
addlocation()	(t)	rans-	method), 428	(ir amstaretster agerarenammi
	•	hod),	addlocations()	(translate.storage.html.htmlunit
484	zang.zangom men	iou),	<i>method</i> ), 438	(iranstate.storage.nimi.nimiini
addlocation()	(t:	rans-	addlocations()	(translate.storage.ical.icalunit
	,	hod),	<i>method</i> ), 443	(ranstare.storage.rear.rearmin
491	iegai.omegai omi men	iou),	addlocations()	(translate.storage.ini.iniunit
addlocation()	(translate.storage.php.ph	nunit	<i>method</i> ), 448	(maistaic.storage.m.mimi
<i>method</i> ), 549	(translate.storage.prip.pri	Juii	addlocations()	(trans-
	anslate.storage.pocommon.po	ounit		onl10n.I18NextUnit method),
method), 555			453	
	nslate.storage.poxliff.PoXlif	fUnit	addlocations()	(trans-
method), 563				$onl 10 n. Js on Ne sted Unit\ method),$
addlocation()	,	rans-	460	
	operties.propunit meti	hod),	addlocations()	(trans-
582				onl10n.JsonUnit method),
addlocation()	(translate.storage.pypo.pe	ounit	463	
method), 591			addlocations()	(trans-
addlocation() <i>method</i> ), 597	(translate.storage.qm.qn	nunit	late.storage.js method), 468	onl10n.WebExtensionJsonUnit
addlocation()	(translate.storage.qph.Qph	ıUnit	addlocations()	(translate.storage.lisa.LISAunit
method), 602			method), 473	
addlocation() (tra	nslate.storage.rc.rcunit meti	hod),	addlocations()	(translate.storage.mo.mounit
607			method), 479	
addlocation()	(tr	rans-	addlocations()	(trans-
late.storage.su 621	btitles.SubtitleUnit meth	hod),	late.storage.m 484	ozilla_lang.LangUnit method),
addlocation()	(translate.storage.tbx.tb.	yunit	addlocations()	(trans-
<i>method</i> ), 626	(iransiaic.siorage.tox.to.	<i>xuiiii</i>		negat.OmegaTUnit method),
addlocation()	(translate.storage.tiki.Tik	i∐nit	491	memou),
<i>method</i> ), 631	(iransiaic.siorage.iiki.1iki	iOnii	addlocations()	(translate.storage.php.phpunit
addlocation()	(translate.storage.tmx.tm.	vunit	<i>method</i> ), 549	(translate.storage.php.phpunii
<i>method</i> ), 636	(Hammune.storage.HIM.HIII	auitti	addlocations()	(trans-
	nslate.storage.trados.Trados	:IInit		ocommon.pounit method),
<i>method</i> ), 640	nome.sioruge.iruuos.11uuos	,	555	memou),
* *	nslate.storage.ts2.tsunit meti	hod)	addlocations()	(trans-
647		,		oxliff.PoXliffUnit method),
addlocation()	(translate.storage.txt.Tx	tUnit	563	,
* *	, , , , , , , , , , , , , , , , , , , ,			

addlocations() (trans-	addnote() (translate.storage.lisa.LISAunit method),
late.storage.properties.propunit method),	473
582	addnote() (translate.storage.mo.mounit method), 480
addlocations() (translate.storage.pypo.pounit	addnote() (translate.storage.mozilla_lang.LangUnit
method), 591	method), 484
addlocations() (translate.storage.qm.qmunit	addnote() (translate.storage.omegat.OmegaTUnit
method), 597	method), 491
addlocations() (translate.storage.qph.QphUnit	addnote() (translate.storage.php.phpunit method),
method), 602	549
addlocations() (translate.storage.rc.rcunit	addnote() (translate.storage.pocommon.pounit
method), 608	method), 555
addlocations() (trans-	addnote() (translate.storage.poxliff.PoXliffUnit
late.storage.subtitles.SubtitleUnit method),	method), 564
621	addnote() (translate.storage.properties.propunit
addlocations() (translate.storage.tbx.tbxunit	method), 582
method), 626	addnote() (translate.storage.pypo.pounit method),
addlocations() (translate.storage.tiki.TikiUnit	591
method), 631	addnote() (translate.storage.qm.qmunit method), 597
addlocations() (translate.storage.tmx.tmxunit	addnote() (translate.storage.qph.QphUnit method),
method), 636	603
addlocations() (trans-	addnote() (translate.storage.rc.rcunit method), 608
late.storage.trados.TradosUnit method),	addnote() (translate.storage.subtitles.SubtitleUnit
641	method), 621
addlocations() (translate.storage.ts2.tsunit	addnote() (translate.storage.tbx.tbxunit method), 626
method), 648	addnote() (translate.storage.tiki.TikiUnit method),
addlocations() (translate.storage.txt.TxtUnit	632
method), 653	addnote() (translate.storage.tmx.tmxunit method),
addlocations() (translate.storage.utx.UtxUnit	637
method), 658	addnote() (translate.storage.trados.TradosUnit
addlocations() (trans-	method), 641
late.storage.wordfast.WordfastUnit method),	addnote() (translate.storage.ts2.tsunit method), 648
669	addnote() (translate.storage.txt.TxtUnit method), 653
addlocations() (translate.storage.xliff.xliffunit	addnote() (translate.storage.utx.UtxUnit method),
method), 675	659
addnote() (translate.storage.base.TranslationUnit	addnote() (translate.storage.wordfast.WordfastUnit
method), 410	method), 669
addnote() (translate.storage.catkeys.CatkeysUnit	addnote() (translate.storage.xliff.xliffunit method),
method), 417	675
addnote() (translate.storage.csvl10n.csvunit method),	addplural() (translate.storage.poxliff.PoXliffFile
422	method), 560
addnote() (translate.storage.dtd.dtdunit method), 428	addsourceunit() (trans-
addnote() (translate.storage.html.htmlunit method),	late.storage.base.TranslationStore method),
438	408
addnote() (translate.storage.ical.icalunit method),	addsourceunit() (trans-
443	late.storage.catkeys.CatkeysFile method),
addnote() (translate.storage.ini.iniunit method), 448	415
addnote() (translate.storage.jsonl10n.118NextUnit	addsourceunit() (translate.storage.csvl10n.csvfile
method), 454	method), 420
addnote() (translate.storage.jsonl10n.JsonNestedUnit	addsourceunit() (translate.storage.dtd.dtdfile
method), 460	method), 426
addnote() (translate.storage.jsonl10n.JsonUnit	addsourceunit() (translate.storage.html.htmlfile
method), 463	method), 435
addnote() (translate.storage.jsonl10n.WebExtensionJso	
method), 468	late.storage.html.POHTMLParser method),
<i>momowy</i> , 100	memou),

432			metho	od), 588		
addsourceunit() <i>method</i> ),441	(translate.storage.i		addsourceur <i>methe</i>	nit() od),595	(translate.storage.	.qm.qmfile
addsourceunit() <i>method</i> ), 446	(translate.storage	e.ini.inifile	addsourceur	pod, $600$	(translate.storage.qp	h.QphFile
addsourceunit()		(trans-	addsourceu		(translate.storag	ge.rc.rcfile
	nl10n.I18NextFile	method),		od), 606	(*	,
452		,,	addsourceu			(trans-
addsourceunit()		(trans-			otitles.AdvSubStationA	`
	nl10n.JsonFile method	`		(od), 612	THE SHARE THE STREET OF THE STREET	upricii iic
addsourceunit()	iii on. 3 soni ne memo	(trans-	addsourceu			(trans-
	nl10n.JsonNestedFile				otitles.MicroDVDFile	method),
458	iii on. 3 som vesicai ne	memou),	614	iorage.suc	illes.melob v b1 lie	memou),
addsourceunit()		(trans-	addsourceu	ni+()		(trans-
	nllOn WahErtansian L				htitles CubPinFile	method),
	nl10n.WebExtensionJ	sonrue	616	iorage.sui	btitles.SubRipFile	meinoa),
method), 466	(tuanalata ataugaa lie	a I ICAGIa				(tuana
addsourceunit()	(translate.storage.lis	а.ызајне	addsourceu		atitlas CulaCtatian Almh	(trans-
method), 471	(4 1 4	CI		-	ptitles.SubStationAlpho	ағие
addsourceunit()	(translate.storage	.то.толіе		od), 617		
method), 476		,	addsourceu			(trans-
addsourceunit()		(trans-		torage.sul	otitles.SubtitleFile	method),
_	zilla_lang.LangStore	method),	619			
482			addsourceu		(translate.storage.	.tbx.tbxfile
addsourceunit()		(trans-		od), 624		
_	egat.OmegaTFile	method),	addsourceu	nit()	(translate.storage.tik	i.TikiStore
488				od), 630		
addsourceunit()		(trans-	addsourceu	nit()	(translate.storage.t	mx.tmxfile
late.storage.om	egat.OmegaTFileTab	method),	metho	od), 634		
489			addsourceu	nit()		(trans-
addsourceunit() <i>method</i> ), 547	(translate.storage.p	php.phpfile	late.s 643	torage.tra	dos.TradosTxtTmFile	method),
addsourceunit()		(trans-	addsourceu	nit()	(translate.storag	e.ts2.tsfile
	ommon.pofile method	1), 552		od), 645	`	
addsourceunit()	1 3	(trans-	addsourceu		(translate.storage.	txt.TxtFile
	liff.PoXliffFile method	*		od), 651	(*	
addsourceunit()	33	(trans-	addsourceu		(translate.storage.u	tx.UtxFile
late.storage.pro	perties.javafile	method),		od), 656	(	
575	r	,,	addsourceu			(trans-
addsourceunit()		(trans-			rdfast.WordfastTMFile	`
	perties.javautf8file	method),	667	iorageino	ragasi. Woragasi 1111 w	, memou),
577	per nes.javanig sjite	memou),	addsourceu	nit()	(translate.storage.x	liff ylifffile
addsourceunit()		(trans-		(20d), 672	(transtate.storage.x	ujj.xujjjuc
	perties.joomlafile	method),	addtransla		(translate.storage.t	mv tmvfile
578	рениез. јоониајне	memou),		(2011), 635	(iransiaie.siorage.i	піл.тілјие
addsourceunit()		(trans-	addunit()		te.storage.base.Transl	ation Store
late.storage.pro	nartias propfila	method),	* * *	od), 408	ie.siorage.base.fransi	unonsiore
580	ретиеѕ.ргорјие	memou),	addunit()		slate.storage.catkeys.C	atkove Fila
		(trans		(iransolution), 415	naie.siorage.caikeys.C	uikeyst iie
addsourceunit()	perties.stringsfile	(trans- method),			storage.csvl10n.csvfile	mathad
585	pernes.siringsjue		420		·	
addsourceunit()		(trans-			torage.dtd.dtdfile metl	
	perties.stringsutf8file	method),		(translate	.storage.html.htmlfile	method),
586			435			
addsourceunit()	(translate.storage.p	ypo.pofile	addunit()	(transla	te.storage.html.POHT	MLParser

method), 432	630
addunit() (translate.storage.ical.icalfile method), 441	addunit () (translate.storage.tmx.tmxfile method), 635
addunit() (translate.storage.ini.inifile method), 446	addunit() (translate.storage.trados.TradosTxtTmFile
addunit() (translate.storage.jsonl10n.118NextFile	method), 643
method), 452	addunit () (translate.storage.ts2.tsfile method), 645
addunit() (translate.storage.jsonl10n.JsonFile	addunit () (translate.storage.txt.TxtFile method), 651
method), 456	addunit () (translate.storage.utx.UtxFile method), 656
addunit() (translate.storage.jsonl10n.JsonNestedFile	addunit () (translate.storage.wordfast.WordfastTMFile
method), 458	method), 667
addunit () (translate.storage.jsonl10n.WebExtensionJso	
method), 466	
	AdvSubStationAlphaFile (class in trans-
addunit() (translate.storage.lisa.LISAfile method),	late.storage.subtitles), 612
471	af (class in translate.lang.af), 353
addunit () (translate.storage.mo.mofile method), 476	AltAttrPlaceable (class in trans-
addunit() (translate.storage.mozilla_lang.LangStore	late.storage.placeables.general), 512
method), 482	<pre>alter_length() (translate.lang.af.af class method),</pre>
addunit() (translate.storage.omegat.OmegaTFile	353
method), 488	alter_length() (translate.lang.am.am class
<pre>addunit() (translate.storage.omegat.OmegaTFileTab</pre>	method), 353
method), 489	<pre>alter_length() (translate.lang.ar.ar class method),</pre>
addunit () (translate.storage.php.phpfile method), 547	354
addunit() (translate.storage.pocommon.pofile	<pre>alter_length() (translate.lang.bn.bn class method);</pre>
method), 552	355
addunit() (translate.storage.poxliff.PoXliffFile	alter_length() (translate.lang.code_or.code_or
method), 560	class method), 356
addunit() (translate.storage.properties.javafile	<pre>alter_length() (translate.lang.common.Common</pre>
method), 575	class method), 357
addunit() (translate.storage.properties.javautf8file	<pre>alter_length() (translate.lang.de.de class method),</pre>
method), 577	363
addunit() (translate.storage.properties.joomlafile	<pre>alter_length() (translate.lang.el.el class method),</pre>
method), 578	364
addunit() (translate.storage.properties.propfile	<pre>alter_length() (translate.lang.es.es class method),</pre>
method), 580	364
addunit () (translate.storage.properties.stringsfile	<pre>alter_length() (translate.lang.fa.fa class method)</pre>
method), 585	365
addunit () (translate.storage.properties.stringsutf8file	<pre>alter_length() (translate.lang.fi.fi class method),</pre>
method), 586	366
addunit () (translate.storage.pypo.pofile method), 588	
addunit () (translate.storage.qm.qmfile method), 595	367
addunit() (translate.storage.qph.QphFile method),	<pre>alter_length() (translate.lang.gu.gu class method);</pre>
600	368
addunit () (translate.storage.rc.rcfile method), 606	<pre>alter_length() (translate.lang.he.he class method),</pre>
addunit () (translate.storage.subtitles.AdvSubStationAlp	phaFile 368
method), 612	alter_length() (translate.lang.hi.hi class method),
addunit() (translate.storage.subtitles.MicroDVDFile	369
method), 614	<pre>alter_length() (translate.lang.hy.hy class method),</pre>
addunit() (translate.storage.subtitles.SubRipFile	370
method), 616	<pre>alter_length() (translate.lang.ja.ja class method),</pre>
addunit()(translate.storage.subtitles.SubStationAlphaF	
method), 617	alter_length() (translate.lang.km.km class
addunit() (translate.storage.subtitles.SubtitleFile	method), 372
method), 619	<pre>alter_length() (translate.lang.kn.kn class method),</pre>
addunit () (translate.storage.tbx.tbxfile method), 624	372
addunit() (translate.storage.tiki.TikiStore method),	<pre>alter_length() (translate.lang.ko.ko class method),</pre>

373 alter_length() (translate.lang.ml.ml class method), 374	apply_to_strings() (trans- late.storage.placeables.base.Ept method), 499
alter_length() (translate.lang.mr.mr class method), 375	apply_to_strings() (trans- late.storage.placeables.base.Ex method),
alter_length() (translate.lang.ne.ne class method), 376	507 apply_to_strings() (trans-
alter_length() (translate.lang.pa.pa class method),  377	apply_to_strings() (trans- late.storage.placeables.base.G method), 504
alter_length() (translate.lang.si.si class method), 378	<pre>apply_to_strings() (trans- late.storage.placeables.base.It method),</pre>
<pre>alter_length() (translate.lang.st.st class method),</pre>	502
379 alter_length() (translate.lang.sv.sv class method), 379	apply_to_strings() (trans- late.storage.placeables.base.Ph method), 500
<pre>alter_length() (translate.lang.ta.ta class method),</pre>	apply_to_strings() (trans-
380 alter_length() (translate.lang.te.te class method),	late.storage.placeables.base.Sub method), 510
381	apply_to_strings() (trans-
alter_length() (translate.lang.th.th class method), 382	late.storage.placeables.base.X method), 508
alter_length() (translate.lang.ug.ug class method),	apply_to_strings() (trans-
383 alter_length() (translate.lang.ur.ur class method),	late.storage.placeables.general.AltAttrPlaceable method), 512
384	apply_to_strings() (trans-
<pre>alter_length() (translate.lang.vi.vi class method),</pre>	late.storage.placeables.general.XMLEntityPlaceable method), 513
<pre>alter_length() (translate.lang.zh.zh class method),</pre>	apply_to_strings() (trans-
385	late.storage.placeables.general.XMLTagPlaceable method), 515
am (class in translate.lang.am), 353 ANALYZER_DEFAULT (trans-	apply_to_strings() (trans-
late.search.indexing.CommonIndexer.CommonD attribute), 397	
ANALYZER_EXACT (trans-	apply_to_strings() (trans-
late.search.indexing.CommonIndexer.CommonD attribute), 397	method), 519
ANALYZER_PARTIAL (trans-	apply_to_strings() (trans-
late.search.indexing.CommonIndexer.CommonD attribute), 397	method), 520
ANALYZER_TOKENIZE (trans- late.search.indexing.CommonIndexer.CommonD	apply_to_strings() (trans- Outabase late.storage.placeables.interfaces.ReplacementPlaceable
attribute), 397	method), 522
append_file() (trans-	apply_to_strings() (trans-
late.storage.bundleprojstore.BundleProjectStore method), 413	late.storage.placeables.interfaces.SubflowPlaceable method), 524
append_file() (trans-	apply_to_strings() (trans-
late.storage.projstore.ProjectStore method), 568	late.storage.placeables.strelem.StringElem method), 526
apply_to_strings() (trans-	apply_to_strings() (trans-
late.storage.placeables.base.Bpt method), 497	late.storage.placeables.terminology.TerminologyPlaceable method), 528
apply_to_strings() (trans-	apply_to_strings() (trans-
late.storage.placeables.base.Bx method),	late.storage.placeables.xliff.Bpt method),
505	530

<pre>apply_to_strings() (trans- late.storage.placeables.xliff.Bx method),</pre>	blank() (translate.filters.checks.KdeChecker method), 292
535	blank() (translate.filters.checks.L20nChecker
apply_to_strings() (trans-	method), 298
late.storage.placeables.xliff.Ept method), 532	blank() (translate.filters.checks.LibreOfficeChecker method), 304
apply_to_strings() (trans-	blank() (translate.filters.checks.MinimalChecker
late.storage.placeables.xliff.Ex method),	method), 310
537	blank() (translate.filters.checks.MozillaChecker
apply_to_strings() (trans-	method), 315
late.storage.placeables.xliff.G method), 538	blank() (translate.filters.checks.OpenOfficeChecker
<pre>apply_to_strings() (trans-</pre>	method), 321
late.storage.placeables.xliff.It method), 540	$\verb blank()  & \textit{(translate.filters.checks.ReducedChecker} \\$
apply_to_strings() (trans-	method), 327
late.storage.placeables.xliff.Ph method), 543	blank() (translate.filters.checks.StandardChecker method), 333
apply_to_strings() (trans-	blank() (translate.filters.checks.TermChecker method),
late.storage.placeables.xliff.Sub method),	340
541	bn (class in translate.lang.bn), 355
apply_to_strings() (trans- late.storage.placeables.xliff.UnknownXML	Bpt (class in translate.storage.placeables.base), 497 Bpt (class in translate.storage.placeables.xliff), 530
method), 545	brackets() (translate.filters.checks.CCLicenseChecker
apply_to_strings() (trans-	method), 269
late.storage.placeables.xliff.X method), 533	brackets() (translate.filters.checks.DrupalChecker
applytranslation() (in module trans-	method), 275
late.convert.po2dtd), 246	brackets() (translate.filters.checks.GnomeChecker
applytranslation() (in module trans-	method), 281
late.convert.po2prop), 252	brackets() (translate.filters.checks.IOSChecker
ar (class in translate.lang.ar), 354	method), 287
ArchiveConvertOptionParser (class in trans- late.convert.convert), 233	method), 292
В	brackets() (translate.filters.checks.L20nChecker method), 298
BasePlaceable (class in trans-	brackets()(translate.filters.checks.LibreOfficeChecket
late.storage.placeables.interfaces), 517	method), 304
batchruntests() (in module trans- late.filters.checks), 346	brackets() (translate.filters.checks.MinimalChecker method), 310
	brackets() (translate.filters.checks.MozillaChecker
late.search.indexing.CommonIndexer.CommonD	
method), 397	brackets() (translate.filters.checks.OpenOfficeChecke
begin_transaction() (trans-	method), 321  Physics (charles of the state
method), $400$	Daichakets () (translate.filters.checks.ReducedChecker method), 327
begin_transaction() (trans-	brackets() (translate.filters.checks.StandardChecker method), 333
late.search.indexing.XapianIndexer.XapianDatal	brackets() (translate.filters.checks.TermChecker
method), 403 blank() (translate.filters.checks.CCLicenseChecker	method), 340
blank () (translate.filters.checks.CCLicenseChecker method), 269	build_checkerconfig() (in module trans-
blank() (translate.filters.checks.DrupalChecker	late.filters.pofilter), 351
method), 275	build_idml_store() (in module trans-
blank() (translate.filters.checks.GnomeChecker	late.storage.xml_extract.extract), 678
method), 281	build_store() (in module trans-
blank() (translate.filters.checks.IOSChecker method),	late.storage.xml_extract.extract), 679
287	<pre>build_unit_tree() (in module trans-</pre>

late.storage.xml_extract.unit_tree), 680	class method), 592
	buildfromunit() (translate.storage.qm.qmunit class
late.tools.poconflicts.ConflictOptionParser method), 683	method), 597 buildfromunit() (translate.storage.qph.QphUnit
buildfromunit() (trans-	buildfromunit () (translate.storage.qph.QphUnit class method), 603
•	buildfromunit() (translate.storage.rc.rcunit class
method), 410	method), 608
buildfromunit() (trans-	buildfromunit() (trans-
late.storage.catkeys.CatkeysUnit class method), 417	late.storage.subtitles.SubtitleUnit class method), 621
buildfromunit() (translate.storage.csvl10n.csvunit	buildfromunit() (translate.storage.tbx.tbxunit class
class method), 422	method), 626
buildfromunit() (translate.storage.dtd.dtdunit class method), 428	buildfromunit() (translate.storage.tiki.TikiUnit class method), 632
<pre>buildfromunit() (translate.storage.html.htmlunit</pre>	buildfromunit() (translate.storage.tmx.tmxunit
class method), 438	class method), 637
<pre>buildfromunit() (translate.storage.ical.icalunit</pre>	•
class method), 443	late.storage.trados.TradosUnit class method),
buildfromunit() (translate.storage.ini.iniunit class	641
<pre>method), 448 buildfromunit() (trans-</pre>	buildfromunit() (translate.storage.ts2.tsunit class
•	<pre>method), 648 buildfromunit() (translate.storage.txt.TxtUnit</pre>
method), 454	class method), 653
	buildfromunit() (translate.storage.utx.UtxUnit
late.storage.jsonl10n.JsonNestedUnit class	class method), 659
method), 460	buildfromunit() (trans-
buildfromunit() (trans-	late.storage.wordfast.WordfastUnit class
late.storage.jsonl10n.JsonUnit class method),	method), 669
463	buildfromunit() (translate.storage.xliff.xliffunit
buildfromunit() (trans-	class method), 675
late.storage.jsonl10n.WebExtensionJsonUnit class method), 468	buildtag() (translate.storage.html.htmlfile method), 435
buildfromunit() (translate.storage.lisa.LISAunit class method), 473	<pre>buildtag() (translate.storage.html.POHTMLParser     method), 432</pre>
buildfromunit() (translate.storage.mo.mounit class method), 480	buildunits() (translate.search.match.matcher method), 405
	buildunits() (trans-
late.storage.mozilla_lang.LangUnit class method), 485	late.search.match.terminologymatcher method), 406
buildfromunit() (trans-	BundleProjectStore (class in trans-
late.storage.omegat.OmegaTUnit class method), 491	late.storage.bundleprojstore), 413 Bx (class in translate.storage.placeables.base), 505
buildfromunit() (translate.storage.php.phpunit	Bx (class in translate.storage.placeables.xliff), 535
<pre>class method), 549 buildfromunit() (trans-</pre>	bzr (class in translate.storage.versioncontrol.bzr), 663
late.storage.pocommon.pounit class method),	C
555	
buildfromunit() (trans-	calcstats_old() (in module trans- late.tools.pocount), 686
late.storage.poxliff.PoXliffUnit class method),	cancel_transaction() (trans-
564	late.search.indexing.CommonIndexer.CommonDatabase
buildfromunit() (trans-	method), 397
late.storage.properties.propunit class method),	cancel_transaction() (trans-
582	late.search.indexing.PyLuceneIndexer.PyLuceneDatabase
<pre>buildfromunit() (translate.storage.pypo.pounit</pre>	method), 400

capitalize() (translate.misc.multistring.multistring method), 387 capistart() (translate.lang.afafclass method), 353 capistart() (translate.lang.am.am class method), 354 capistart() (translate.lang.am.bm.bclass method), 355 capistart() (translate.lang.am.bm.bclass method), 356 capistart() (translate.lang.am.bm.bclass method), 356 capistart() (translate.lang.acomonon.Common class method), 357 capistart() (translate.lang.acomonon.Common class method), 357 capistart() (translate.lang.acomonon.Common class method), 356 capistart() (translate.lang.acomonon.Common class method), 364 capistart() (translate.lang.acomonon.Common class method), 366 capistart() (translate.lang.afa class method), 366 capistart() (translate.lang.afifr class method), 366 capistart() (translate.lang.afifr class method), 366 capistart() (translate.lang.afifr class method), 367 capistart() (translate.lang.afifr class method), 368 capistart() (translate.lang.afifr class method), 369 capistart() (translate.lang.afine class method), 370 capistart() (translate.lang.afine class method), 373 capistart() (translate.lang.afine class method), 374 capistart() (translate.lang.afine class method), 375 capistart() (translate.lang.afine class method), 386 capistart() (translate.lang.afine class method), 376 capistart() (translate.lang.afine class method), 376 capistart() (translate.lang.afine class method), 376 capistart() (translate.lang.a	cancel_transaction() (trans- late.search.indexing.XapianIndexer.XapianDatab method), 403	CatkeysUnit(classi caseLicenseChecker 269		
capsstart() (translate.lang.am.am class method), 353 capsstart() (translate.lang.ar.ar class method), 354 capsstart() (translate.lang.ar.ar class method), 355 capsstart() (translate.lang.code_or.code_or class method), 356 capsstart() (translate.lang.common.Common class method), 357 capsstart() (translate.lang.common.Common class method), 357 capsstart() (translate.lang.e.e.e.e.e.e.e.e.e.e.e.e.e.e.e.e.e.e.e	capitalize() (translate.misc.multistring.multistring	center() (tran	slate.misc.multistring.mu	ltistring
capsstart() (translate.lang.am.am class method), 354 capsstart() (translate.lang.corar class method), 355 capsstart() (translate.lang.common.Common class method), 356 capsstart() (translate.lang.common.Common class method), 356 capsstart() (translate.lang.g.e.es class method), 366 capsstart() (translate.lang.g.fi, class method), 366 capsstart() (translate.lang.fi, class method), 366 capsstart() (translate.lang.fi, class method), 367 capsstart() (translate.lang.g.u.g.u class method), 368 capsstart() (translate.lang.g.u.g.u class method), 369 capsstart() (translate.lang.m.k.m class method), 369 capsstart() (translate.lang.g.i.p.a class method), 370 capsstart() (translate.lang.g.i.m.m class method), 373 capsstart() (translate.lang.m.m.m class method), 373 capsstart() (translate.lang.g.n.m.m class method), 373 capsstart() (translate.lang.g.n.m.m class method), 373 capsstart() (translate.lang.m.m.m class method), 373 capsstart() (translate.lang.m.m.m class method), 373 capsstart() (translate.lang.g.n.s.st class method), 373 capsstart() (translate.lang.g.n.s.st class method), 373 capsstart() (translate.lang.g.n.s.st class method), 376 capsstart() (translate.lang.g.n.s.st class method), 377 capsstart() (translate.lang.g.n.s.st class method), 378 capsstart() (translate.lang.g.n.s.st class method), 378 capsstart() (translate.lang.g.n.s.st class method), 380 capsstart() (translate.lang.g.n.s.st class method), 380 capsstart() (translate.lang.g.n.st class method), 381 capsstart() (translate.lang.g.n.st class method), 382 capsstart() (translate.la		, · · · · · · · · · · · · · · · · · · ·	(1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.	
capstart() (translate.lang.a.ar class method), 355 capstart() (translate.lang.bn.hn class method), 355 capstart() (translate.lang.common.Common class method), 356 capstart() (translate.lang.e.dec class method), 363 capstart() (translate.lang.e.dec class method), 364 capstart() (translate.lang.g.fa.fa class method), 364 capstart() (translate.lang.g.fa.fa class method), 366 capstart() (translate.lang.fa.fa class method), 367 capstart() (translate.lang.fa.fa class method), 368 capstart() (translate.lang.fa.fa class method), 366 capstart() (translate.lang.fa.fa class method), 369 capstart() (translate.lang.fa.fa class method), 373 capstart() (translate.lang.fa.fa class method), 373 capstart() (translate.lang.fa.fa class method), 373 capstart() (translate.lang.fa.fa class method), 376 capstar				trans-
capsstart() (translate.lang.arar class method), 354 capsstart() (translate.lang.come code_or class method), 355 capsstart() (translate.lang.come class method), 356 capsstart() (translate.lang.de.de class method), 364 capsstart() (translate.lang.fi.el.ass method), 364 capsstart() (translate.lang.fi.el.ass method), 364 capsstart() (translate.lang.fi.el.ass method), 366 capsstart() (translate.lang.fi.el.ass method), 368 capsstart() (translate.lang.fi.el.ass method), 370 capsstart() (translate.lang.fi.el.ass method), 371 capsstart() (translate.lang.fi.el.ass method), 373 capsstart() (translate.lang.fi.el.ass method), 373 capsstart() (translate.lang.m.m. class method), 374 capsstart() (translate.lang.fi.el.ass method), 375 capsstart() (translate.lang.fi.el.ass method), 376 capsstart() (translate.lang.fi.el.ass method), 380 capsstart() (translate.lang.fi.el.ass metho				
capstart () (translate.lang.nbn.class method), 355 capstart () (translate.lang.common.Common class method), 354 capstart () (translate.lang.de.de class method), 363 capstart () (translate.lang.de.de class method), 364 capstart () (translate.lang.fa.fa class method), 364 capstart () (translate.lang.fa.fa class method), 365 capstart () (translate.lang.fi.fr class method), 366 capstart () (translate.lang.m.fi.fr class method), 366 capstart () (translate.lang.fi.fr class method), 366 capstart () (translate.lang.fi.fr class method), 369 capstart () (translate.lang.fi.fr class method), 379 capstart () (translate.lang.fi.fr class method), 373 capstart () (translate.lang.fi.fr class method), 374 capstart () (translate.lang.fi.fr class method), 375 capstart () (translate.lang.fi.fr class method), 376 capstart () (translate.lang.fi.fr class method), 378 capstart () (translate.lang.fi.fr class method), 378 capstart () (translate.lang.fi.fr class method), 378 capstart () (translate.lang.fi.fr class method), 376 capstart () (translate.l			(translate.lang.af.af	class
capstart() (translate.lang.common.Common class method), 356 apsstart() (translate.lang.de.de class method), 364 apsstart() (translate.lang.de.de class method), 364 apsstart() (translate.lang.de.de class method), 365 apsstart() (translate.lang.fi.el class method), 366 apsstart() (translate.lang.fi.el class method), 366 apsstart() (translate.lang.fi.el class method), 366 apsstart() (translate.lang.ge.de class method), 369 apsstart() (translate.lang.ge.de class method), 369 apsstart() (translate.lang.hi.hi.class method), 370 apsstart() (translate.lang.km.km class method), 371 apsstart() (translate.lang.km.km class method), 372 capsstart() (translate.lang.km.km class method), 373 appsstart() (translate.lang.me.ne class method), 373 appsstart() (translate.lang.me.me class method), 373 appsstart() (translate.lang.me.ne class method), 375 appsstart() (translate.lang.se.se class method), 376 appsstart() (translate.lang.me.me class method), 376 appsstart() (translate.lang.se.se class method), 377 appsstart() (translate.lang.me.me class method), 378 appsstart() (translate.lang.me.me class method), 378 appsstart() (translate.lang.se.se class method), 378 appsstart() (translate.lang.se.se class method), 378 appsstart() (translate.lang.me.me class method), 378 appsstart() (translate.lang.se.se class method), 379 appsstart() (translate.lang.se.se class		, · · · · · · · · · · · · · · · · · · ·	, , ,	
capstart() (translate.lang.common.Common class method), 355 capstart() (translate.lang.el.el class method), 364 capstart() (translate.lang.el.el class method), 364 capstart() (translate.lang.el.el class method), 366 capstart() (translate.lang.fl.g.a class method), 367 capstart() (translate.lang.fl.g.a class method), 369 capstart() (translate.lang.fl.g.a class method), 369 capstart() (translate.lang.fl.g.a class method), 369 capstart() (translate.lang.fl.g.a class method), 371 capstart() (translate.lang.g.h.in class method), 372 capstart() (translate.lang.m.m.rclass method), 373 capstart() (translate.lang.m.m.rclass method), 374 capstart() (translate.lang.m.m.rclass method), 375 capstart() (translate.lang.g.n.e. class method), 376 capstart() (translate.lang.g.n.e. class method), 377 capstart() (translate.lang.g.n.e. class method), 378 capstart() (translate.lang.g.n.e. class method), 378 capstart() (translate.lang.g.n.e. class method), 378 capstart() (translate.lang.g.n.e. class method), 380 capstart() (translate.lang.g.n.e.	- · · · · · · · · · · · · · · · · · · ·		(translate.lang.am.am	class
capsstart() (translate.lang.common.Common class method), 357 capsstart() (translate.lang.el.el class method), 364 capsstart() (translate.lang.el.el class method), 365 capsstart() (translate.lang.gl.el.el class method), 366 capsstart() (translate.lang.gl.el.el.el.eng.fi.el.ass method), 366 capsstart() (translate.lang.gl.el.el.el.eng.gl.el.el.el.eng.gl.el.el.el.eng.gl.el.el.el.eng.gl.el.el.el.eng.gl.el.el.el.eng.gl.el.el.el.eng.gl.el.el.el.el.eng.gl.el.el.el.el.eng.gl.el.el.el.el.eng.gl.el.el.el.el.eng.gl.el.el.el.el.eng.el.el.el.el.eng.el.el.el.el.eng.el.el.el.el.eng.gl.el.el.el.el.eng.el.el.el.el.eng.el.el.el.el.el.el.el.el.el.el.el.el.el.				_
capstart() (translate.lang.elecleclass method), 364 capstart() (translate.lang.es.es class method), 364 capstart() (translate.lang.fi.fi class method), 366 capstart() (translate.lang.g.fi.fi class method), 366 capstart() (translate.lang.g.h.ini class method), 369 capstart() (translate.lang.g.h.ini class method), 370 capstart() (translate.lang.g.h.ini class method), 371 capstart() (translate.lang.g.h.ini class method), 372 capstart() (translate.lang.g.h.ini class method), 373 capstart() (translate.lang.g.h.ini class method), 374 capstart() (translate.lang.g.h.ini class method), 375 capstart() (translate.lang.g.h.ini class method), 375 capstart() (translate.lang.g.h.ini class method), 375 capstart() (translate.lang.g.h.ini class method), 376 capsta			(translate.lang.ar.ar	class
capsstart() (translate.lang.de.de class method), 364 capsstart() (translate.lang.ee.let class method), 365 capsstart() (translate.lang.fi,fi class method), 365 capsstart() (translate.lang.fi,fi class method), 367 capsstart() (translate.lang.fi,fi class method), 367 capsstart() (translate.lang.gh.fi class method), 368 capsstart() (translate.lang.gh.eh e class method), 369 capsstart() (translate.lang.gh.eh e class method), 369 capsstart() (translate.lang.hy.hy class method), 369 capsstart() (translate.lang.hy.hy class method), 369 capsstart() (translate.lang.gh.m.km class method), 370 capsstart() (translate.lang.km.km class method), 370 capsstart() (translate.lang.km.km class method), 373 capsstart() (translate.lang.m.m.mr class method), 373 capsstart() (translate.lang.m.m.mr class method), 373 capsstart() (translate.lang.gh.en e class method), 374 capsstart() (translate.lang.si.si class method), 375 capsstart() (translate.lang.si.si class method), 376 capsstart() (translate.lang.si.si class method), 378 capsstart() (translate.lang.si.si class method), 380 capsstart() (translat				_
capsstart() (translate.lang.e.ea class method), 364 capsstart() (translate.lang.g.fa class method), 365 capsstart() (translate.lang.fi.fi class method), 366 capsstart() (translate.lang.fi.fi class method), 367 capsstart() (translate.lang.fi.fi class method), 368 capsstart() (translate.lang.fi.fi class method), 368 capsstart() (translate.lang.fi.fi class method), 369 capsstart() (translate.lang.hi.hi class method), 369 capsstart() (translate.lang.g.hi.hi class method), 369 capsstart() (translate.lang.g.hi.hi class method), 370 capsstart() (translate.lang.g.hi.hi class method), 371 capsstart() (translate.lang.g.hi.hi class method), 373 capsstart() (translate.lang.m.mr class method), 374 capsstart() (translate.lang.g.hi.mi class method), 375 capsstart() (translate.lang.g.hi.mi class method), 375 capsstart() (translate.lang.g.hi.mi class method), 376 capsstart() (translate.lang.g.hi.mi class method), 376 capsstart() (translate.lang.g.hi.mi class method), 375 capsstart() (translate.lang.g.hi.mi class method), 376 capsstart() (translate.lang.g.hi.mi class method), 376 capsstart() (translate.lang.g.hi.mi class method), 378 capsstart() (translate.lang.g.hi.hi class method), 380 capsstart() (translate.lang.g.hi.hi class method), 380 capsstart() (translate.lang.g.hi.hi class method), 380 capsstart() (translate.lang.g.hi.hi class method), 381 capsstart() (translate.lang.g.hi			(translate.lang.bn.bn	class
capsstart() (translate.lang.fa.fa class method), 365 capsstart() (translate.lang.fi.fa class method), 366 capsstart() (translate.lang.fi.fr class method), 367 capsstart() (translate.lang.fi.fr class method), 368 capsstart() (translate.lang.g.n.gu class method), 369 capsstart() (translate.lang.h.i.hi class method), 369 capsstart() (translate.lang.h.j.h.y class method), 370 capsstart() (translate.lang.h.j.h.y class method), 370 capsstart() (translate.lang.h.j.h.y class method), 371 capsstart() (translate.lang.km.km class method), 371 capsstart() (translate.lang.km.km class method), 372 capsstart() (translate.lang.km.km class method), 373 capsstart() (translate.lang.km.km class method), 373 capsstart() (translate.lang.m.m.lml class method), 373 capsstart() (translate.lang.m.m.lml class method), 373 capsstart() (translate.lang.m.m.ml class method), 374 capsstart() (translate.lang.m.e.ne class method), 375 capsstart() (translate.lang.s.s.s class method), 376 capsstart() (translate.lang.s.s.s class method), 378 capsstart() (translate.lang.s.s.s class method), 380 capsstart() (trans		, · · · · · · · · · · · · · · · · · · ·		
capsstart() (translate.lang.fi.fi class method), 365 capsstart() (translate.lang.fi.fi class method), 366 capsstart() (translate.lang.fi.fi class method), 367 capsstart() (translate.lang.g.e.gu.class method), 368 capsstart() (translate.lang.he.he class method), 369 capsstart() (translate.lang.hy.hy.class method), 370 capsstart() (translate.lang.km.km class method), 371 capsstart() (translate.lang.km.km class method), 372 capsstart() (translate.lang.km.km class method), 373 capsstart() (translate.lang.km.km class method), 373 capsstart() (translate.lang.km.km class method), 373 capsstart() (translate.lang.m.mm class method), 374 capsstart() (translate.lang.m.mm class method), 375 capsstart() (translate.lang.m.mm class method), 376 capsstart() (translate.lang.m.mm class method), 377 capsstart() (translate.lang.s.is.i class method), 378 capsstart() (translate.lang.s.is.i class method), 378 capsstart() (translate.lang.s.is.i class method), 378 capsstart() (translate.lang.s.is.i class method), 380 capstart() (translate.lang.s.is.i class method), 380 capstart() (translate.lang.s.is.i class method), 380 capstart() (tr				code_or
capsstart() (translate.lang.fi.fi class method), 366 capsstart() (translate.lang.gu.gu class method), 369 capsstart() (translate.lang.gu.gu class method), 369 capsstart() (translate.lang.hi.hi class method), 369 capsstart() (translate.lang.hi.hi class method), 370 capsstart() (translate.lang.gh.y.hy class method), 371 capsstart() (translate.lang.kn.km class method), 372 capsstart() (translate.lang.kn.km class method), 373 capsstart() (translate.lang.kn.km class method), 373 capsstart() (translate.lang.kn.km class method), 373 capsstart() (translate.lang.ml.ml class method), 373 capsstart() (translate.lang.m.mm class method), 374 capsstart() (translate.lang.m.mm class method), 375 capsstart() (translate.lang.m.mm class method), 376 capsstart() (translate.lang.n.m.m class method), 377 capsstart() (translate.lang.n.m.m class method), 378 capsstart() (translate.lang.si.si class method), 379 capsstart() (translate.lang.si.si class method), 380 capsstart() (translate.lang.si.si class method), 381 capstart() (translate.lang.si.si class method), 382 capstart() (translate.lang.si.si class method), 382 capstart() (translate.lang.si.si class method), 382 categories (translate.lang.si.si cla	capsstart() (translate.lang.es.es class method), 364	class method),	356	
capsstart() (translate.lang.g.u.gu class method), 368 capsstart() (translate.lang.he.class method), 369 capsstart() (translate.lang.hi.hi class method), 369 capsstart() (translate.lang.hi.hi class method), 370 capsstart() (translate.lang.hi.hi class method), 371 capsstart() (translate.lang.km.km class method), 372 capsstart() (translate.lang.km.km class method), 373 capsstart() (translate.lang.km.km class method), 373 capsstart() (translate.lang.mi.ml class method), 374 capsstart() (translate.lang.mi.ml class method), 374 capsstart() (translate.lang.mi.ml class method), 374 capsstart() (translate.lang.mi.ml class method), 375 capsstart() (translate.lang.mi.ml class method), 376 capsstart() (translate.lang.ni.me class method), 377 capsstart() (translate.lang.si.si class method), 377 capsstart() (translate.lang.si.si class method), 378 capsstart() (translate.lang.si.si class method), 380 capsstart() (translate.lang.si.si class method), 380 capsstart() (translate.lang.ti.ti class method), 380 capsstart() (translate.lang.u.cu class method), 382 capsstart() (translate.lang.u.cu class method), 384 capsstart() (translate.lang.u.cu class method), 385 categories (translate.lang.ch.ch class method), 385 categories (translate.lang.ch.ch class method), 386 categories (translate.lang.ch.ch class method), 380 c		<pre>character_iter()</pre>		(trans-
capsstart() (translate.lang.he.he class method), 369 capsstart() (translate.lang.hi.hi class method), 369 capsstart() (translate.lang.hi.hy.hy class method), 370 capsstart() (translate.lang.kn.km class method), 371 capsstart() (translate.lang.kn.km class method), 372 capsstart() (translate.lang.kn.km class method), 373 capsstart() (translate.lang.kn.km class method), 373 capsstart() (translate.lang.m.km class method), 373 capsstart() (translate.lang.m.mr class method), 373 capsstart() (translate.lang.m.mr class method), 374 capsstart() (translate.lang.m.mr class method), 375 capsstart() (translate.lang.n.m.mr class method), 376 capsstart() (translate.lang.pa.pa class method), 376 capsstart() (translate.lang.si.si class method), 377 capsstart() (translate.lang.si.si class method), 378 capsstart() (translate.lang.si.si class method), 380 capsstart() (translate.lang.si.si class method), 382 capsstart() (translate.lang.si.si class method), 382 capsstart() (translate.lang.si.si class method), 384 capsstart() (translate.lang.si.si class method), 382 capsstart() (translate.lang.si.si class method), 382 capsstart() (translate.lang.si.si class method), 382 capsstart() (translate.lang.si.si class method), 384 capstart() (translate.lang.si.si class method), 382 capsstart() (translate.lang.si.si class method), 382 capstart() (translate.lang.si.si class method), 384 capstart() (translate.lang.si.si class method), 384 capstart() (translate.lang.si.si cl	capsstart() (translate.lang.fi.fi class method), 366	late.lang.comn	non.Common class n	nethod),
capsstart() (translate.lang.hi.hi class method), 369 capsstart() (translate.lang.hi.hi class method), 369 capsstart() (translate.lang.hy.hy class method), 370 capsstart() (translate.lang.km.km class method), 371 capsstart() (translate.lang.km.km class method), 373 capsstart() (translate.lang.km.km class method), 373 capsstart() (translate.lang.mn.mr class method), 373 capsstart() (translate.lang.mn.mr class method), 374 capsstart() (translate.lang.mn.mr class method), 375 capsstart() (translate.lang.mn.mr class method), 376 capsstart() (translate.lang.mn.mr class method), 377 capsstart() (translate.lang.s.si.class method), 378 capsstart() (translate.lang.s.si.class method), 379 capsstart() (translate.lang.s.s.si class method), 379 capsstart() (translate.lang.s.s.si class method), 380 capsstart() (translate.lang.s.si.si class method), 380 capsstart() (translate.lang.s.si.si class method), 380 capstart() (translate.lang.s.si.si class method), 380 capstar	capsstart() (translate.lang.fr.fr class method), 367	357		
capsstart() (translate.lang.hi.hi class method), 369 capsstart() (translate.lang.ja.ja class method), 370 capsstart() (translate.lang.ja.ja class method), 371 capsstart() (translate.lang.km.km class method), 373 capsstart() (translate.lang.km.km class method), 373 capsstart() (translate.lang.km.km class method), 373 capsstart() (translate.lang.m.mm class method), 374 capsstart() (translate.lang.m.mm class method), 375 capsstart() (translate.lang.ne.ne class method), 376 capsstart() (translate.lang.pa.pa class method), 376 capsstart() (translate.lang.s.is iclass method), 378 capsstart() (translate.lang.s.s.is class method), 380 capsstart() (translate.lang.s.s.is class method), 381 capsstart() (translate.lang.s.t.is class method), 382 capsstart() (translate.lang.s.t.is class method), 381 capsstart() (translate.lang.s.t.is class method), 381 capsstart() (translate.lang.s.t.is class method), 382 capsstart() (translate.lang.s.t.is class method), 381 capsstart() (translate.lang.s.t.is class method), 382 capsstart() (translate.lang.s.t.is class method), 385 catedories (translate.lang.s.t.is class method), 385 categori	capsstart() (translate.lang.gu.gu class method), 368	<pre>character_iter()</pre>	(translate.lang.de.de	class
capsstart() (translate.lang.hy.hy class method), 370 capsstart() (translate.lang.ja.ja class method), 371 capsstart() (translate.lang.km.km class method), 373 capsstart() (translate.lang.km.km class method), 373 capsstart() (translate.lang.km.km class method), 373 capsstart() (translate.lang.m.lm class method), 373 capsstart() (translate.lang.m.lm class method), 374 capsstart() (translate.lang.m.mr class method), 375 capsstart() (translate.lang.p.apa class method), 376 capsstart() (translate.lang.si.si class method), 377 capsstart() (translate.lang.si.si class method), 378 capsstart() (translate.lang.si.si class method), 378 capsstart() (translate.lang.si.si class method), 379 capsstart() (translate.lang.si.si class method), 380 capsstart() (translate.lang.ti.th class method), 380 capsstart() (translate.lang.ti.th class method), 380 capsstart() (translate.lang.ti.th class method), 381 capsstart() (translate.lang.ti.th class method), 384 capsstart() (translate.lang.un.ur class method), 384 capsstart() (translate.lang.si.si class method), 384 capsstart() (translate.lang.ti.th class method), 385 catesories (translate.lang.ti.th class method), 385 categories (translate.lang.ti.th class method), 385 categories (translate.filters.checks.TeeChecker attribute), 339 categories (translate.filters.checks.UnitChecker attribute), 339 categories (translate.filters.checks.UnitChecker attribute), 339 categories (translate.storage.catkeys), 415 categories (translate.storage.catkeys), 415 CatkeysFile (class in translate.storage.catkeys), 415 CatkeysFile (class	capsstart() (translate.lang.he.he class method), 369	method), 363		
capsstart() (translate.lang.hy.hy class method), 370 capsstart() (translate.lang.ja.ja class method), 371 capsstart() (translate.lang.km.km class method), 373 capsstart() (translate.lang.km.km class method), 373 capsstart() (translate.lang.km.km class method), 373 capsstart() (translate.lang.m.lm class method), 373 capsstart() (translate.lang.m.lm class method), 374 capsstart() (translate.lang.m.mr class method), 375 capsstart() (translate.lang.p.apa class method), 376 capsstart() (translate.lang.si.si class method), 377 capsstart() (translate.lang.si.si class method), 378 capsstart() (translate.lang.si.si class method), 378 capsstart() (translate.lang.si.si class method), 379 capsstart() (translate.lang.si.si class method), 380 capsstart() (translate.lang.ti.th class method), 380 capsstart() (translate.lang.ti.th class method), 380 capsstart() (translate.lang.ti.th class method), 381 capsstart() (translate.lang.ti.th class method), 384 capsstart() (translate.lang.un.ur class method), 384 capsstart() (translate.lang.si.si class method), 384 capsstart() (translate.lang.ti.th class method), 385 catesories (translate.lang.ti.th class method), 385 categories (translate.lang.ti.th class method), 385 categories (translate.filters.checks.TeeChecker attribute), 339 categories (translate.filters.checks.UnitChecker attribute), 339 categories (translate.filters.checks.UnitChecker attribute), 339 categories (translate.storage.catkeys), 415 categories (translate.storage.catkeys), 415 CatkeysFile (class in translate.storage.catkeys), 415 CatkeysFile (class	capsstart() (translate.lang.hi.hi class method), 369	<pre>character_iter()</pre>	(translate.lang.el.el	class
capsstart () (translate.lang.ja.ja.class method), 371 capsstart () (translate.lang.km.km class method), 373 capsstart () (translate.lang.kn.kn class method), 373 capsstart () (translate.lang.m.mm class method), 374 capsstart () (translate.lang.m.mm class method), 375 capsstart () (translate.lang.m.mm class method), 376 capsstart () (translate.lang.pa.pa class method), 376 capsstart () (translate.lang.si.si class method), 377 capsstart () (translate.lang.si.si class method), 378 capsstart () (translate.lang.si.si class method), 379 capsstart () (translate.lang.si.si class method), 379 capsstart () (translate.lang.si.si class method), 380 capsstart () (translate.lang.si.si class method), 381 capstart () (translate.lang.si.si class method), 380 capstart () (translate.lang.si.si class method), 381 capstart () (translate.lang.si.si class method), 381		method), 364		
capsstart() (translate.lang.kn.km class method), 372 capsstart() (translate.lang.kn.kn class method), 373 capsstart() (translate.lang.ko.ko class method), 373 capsstart() (translate.lang.mn.mt class method), 374 capsstart() (translate.lang.mn.mt class method), 375 capsstart() (translate.lang.mn.mt class method), 375 capsstart() (translate.lang.ne.ne class method), 376 capsstart() (translate.lang.pap class method), 377 capsstart() (translate.lang.si.si class method), 378 capsstart() (translate.lang.si.si class method), 380 capstart() (translate.lang.si.si class method), 380 capstart() (translate.lang.si.si class method), 381 capstart() (translate.lang.si.si cla	- · · · · · · · · · · · · · · · · · · ·	, · · · · · · · · · · · · · · · · · · ·	(translate.lang.es.es	class
capstart() (translate.lang.kn.kn class method), 373 capsstart() (translate.lang.m.mr class method), 374 capsstart() (translate.lang.m.mr class method), 375 capsstart() (translate.lang.m.mr class method), 376 capsstart() (translate.lang.m.mr class method), 376 capsstart() (translate.lang.ne.ne class method), 376 capsstart() (translate.lang.s.i.si class method), 377 capsstart() (translate.lang.s.i.si class method), 378 capsstart() (translate.lang.s.i.si class method), 379 capsstart() (translate.lang.s.i.si class method), 380 capsstart() (translate.lang.s.i.si class method), 381 capsstart() (translate.lang.s.i.si class method), 381 capsstart() (translate.lang.s.i.si class method), 382 capsstart() (translate.lang.s.i.si class method), 384 capsstart() (translate.lang.s.i.si class method), 384 capst	-		(	
capsstart () (translate.lang.kn.kn class method), 373 capsstart () (translate.lang.ko.ko class method), 373 capsstart () (translate.lang.m.mr class method), 375 capsstart () (translate.lang.m.mr class method), 376 capsstart () (translate.lang.ne.ne class method), 376 capsstart () (translate.lang.ne.ne class method), 376 capsstart () (translate.lang.si.si class method), 377 capsstart () (translate.lang.si.si class method), 378 capsstart () (translate.lang.si.si class method), 378 capsstart () (translate.lang.si.si class method), 380 capsstart () (translate.lang.st.st class method), 380 capsstart () (translate.lang.st.st class method), 380 capsstart () (translate.lang.st.st class method), 381 capsstart () (translate.lang.st.ta class method), 381 capsstart () (translate.lang.st.th class method), 382 capsstart () (translate.lang.th.th class method), 382 capsstart () (translate.lang.us.us class method), 384 capsstart () (translate.lang.st.tr class method), 385 catedories (translate.lang.st.tr class method), 385 catedories (translate.ling.st.tr class method), 385 catedories (translate.filters.checks.TeeChecker attribute), 346 CatkeysDialect (class in translate.storage.catkeys), 415 CatkeysHeader (class in translate.storage.catkeys), 415 CatkeysHeader (class in translate.storage.catkeys), 415			(translate.lang.fa.fa	class
capsstart() (translate.lang.m.ml class method), 373 capsstart() (translate.lang.m.mr class method), 375 capsstart() (translate.lang.m.me class method), 376 capsstart() (translate.lang.p.a.pa class method), 376 capsstart() (translate.lang.p.a.pa class method), 377 capsstart() (translate.lang.si.si class method), 378 capsstart() (translate.lang.si.si class method), 378 capsstart() (translate.lang.si.si class method), 378 capsstart() (translate.lang.si.si class method), 379 capsstart() (translate.lang.si.si class method), 380 capsstart() (translate.lang.si.si class method), 381 capsstart() (translate.lang.si.si class method), 382 capsstart() (translate.lang.si.si class method), 382 capsstart() (translate.lang.si.si class method), 382 capsstart() (translate.lang.si.si class method), 380 capsstart() (translate.lang.si.si class method), 381 capsstart() (translate.lang.si.si class method), 382 capsstart() (translate.lang.si.si class method), 382 capsstart() (translate.lang.si.si class method), 382 capsstart() (translate.lang.si.si class method), 384 capsstart() (translate.lang.si.si class method), 384 capsstart() (translate.lang.si.si class method), 384 capsstart() (translate.lang.si.si class method), 380 character_iter() (translate.lang.hi.hi class method), 370 character_iter() (translate.lang.si.si class method), 371 character_iter() (translate.lang.si.si class method), 372 character_iter() (translate.lang.si.si class method), 373 character_iter() (translate.lang.si.si class method			(	
capsstart() (translate.lang.mr.mr class method), 374 capsstart() (translate.lang.mr.mr class method), 375 capsstart() (translate.lang.ne.ne class method), 376 capsstart() (translate.lang.pa.pa class method), 377 capsstart() (translate.lang.si.si class method), 378 capsstart() (translate.lang.si.si class method), 378 capsstart() (translate.lang.sv.sv class method), 379 capsstart() (translate.lang.sv.sv class method), 380 capsstart() (translate.lang.ta.ta class method), 380 capsstart() (translate.lang.ta.ta class method), 380 capsstart() (translate.lang.ta.ta class method), 381 capsstart() (translate.lang.us.us class method), 382 capsstart() (translate.lang.us.us class method), 382 capsstart() (translate.lang.us.us class method), 382 capsstart() (translate.lang.us.us class method), 383 capsstart() (translate.lang.us.us class method), 384 capsstart() (translate.lang.us.us class method), 384 capsstart() (translate.lang.us.us class method), 384 capsstart() (translate.lang.us.us class method), 385 CatchStringOutput (class in translate.lang.ta.ta class method), 385 Categories (translate.filters.checks.TeeChecker attribute), 339 categories (translate.filters.checks.UnitChecker attribute), 346 CatkeysDialect (class in translate.storage.catkeys), 415 CatkeysHeader (class in translate.storage.catkeys), 415			(translate.lang.fi.fi	class
capsstart() (translate.lang.mr.mr class method), 375 capsstart() (translate.lang.ne.ne class method), 376 capsstart() (translate.lang.pa.pa class method), 377 capsstart() (translate.lang.si.si class method), 378 capsstart() (translate.lang.si.si class method), 378 capsstart() (translate.lang.si.si class method), 379 capsstart() (translate.lang.si.si class method), 380 capsstart() (translate.lang.sv.sv class method), 380 capsstart() (translate.lang.ta.ta class method), 380 capsstart() (translate.lang.ta.ta class method), 381 capsstart() (translate.lang.us.us class method), 382 capsstart() (translate.lang.us.us class method), 382 capsstart() (translate.lang.us.us class method), 383 capsstart() (translate.lang.us.us class method), 384 capsstart() (translate.lang.us.us class method), 385 catecher_iter() (translate.lang.ja.ja class method), 370 character_iter() (translate.lang.km.km class method), 372 character_iter() (translate.lang.km.km class method), 373 character_iter() (translate.lang.kn.kn class method), 373 character_iter() (translate.lang.kn.kn class method), 373 character_iter() (translate.lang.mi.mi class method), 373 character_iter() (translate.lang.mi.mi class method), 375 CatkeysHeader (class in translate.storage.catkeys), 415			(ir aristarentario, grigi	Creass
capsstart() (translate.lang.ne.ne class method), 376 capsstart() (translate.lang.pa.pa class method), 378 capsstart() (translate.lang.si.si class method), 378 capsstart() (translate.lang.si.st class method), 379 capsstart() (translate.lang.si.st class method), 380 capsstart() (translate.lang.si.st class method), 381 capsstart() (translate.lang.si.st class method), 382 capsstart() (translate.lang.si.st class method), 382 capsstart() (translate.lang.si.st class method), 383 capsstart() (translate.lang.si.st class method), 384 capstart() (translate.lang.si.st cla			(translate lang fr fr	class
capsstart() (translate.lang.pa.pa class method), 378 capsstart() (translate.lang.si.si class method), 378 capsstart() (translate.lang.st.st class method), 379 capsstart() (translate.lang.st.st class method), 380 capsstart() (translate.lang.sv.sv class method), 380 capsstart() (translate.lang.ta.ta class method), 380 capsstart() (translate.lang.te.te class method), 381 capsstart() (translate.lang.th.th class method), 382 capsstart() (translate.lang.ug.ug class method), 382 capsstart() (translate.lang.ug.us class method), 383 capsstart() (translate.lang.unur class method), 384 capsstart() (translate.lang.vi.vi class method), 385 CatchStringOutput (class in translate.storage.catkeys), 399 categories (translate.filters.checks.UnitChecker attibute), 346 CatkeysDialect (class in translate.storage.catkeys), 415 CatkeysHeader (class in translate.storage.catkeys), 415			(transiate.tang.jr.jr	ciuss
capsstart() (translate.lang.si.si class method), 378 capsstart() (translate.lang.st.st class method), 379 capsstart() (translate.lang.sv.sv class method), 380 capsstart() (translate.lang.ta.ta class method), 380 capsstart() (translate.lang.te.te class method), 381 capsstart() (translate.lang.th.th class method), 382 capsstart() (translate.lang.ug.ug class method), 382 capsstart() (translate.lang.ug.ug class method), 383 capsstart() (translate.lang.ur.ur class method), 384 capsstart() (translate.lang.vi.vi class method), 384 capsstart() (translate.lang.ur.ur class method), 384 capsstart() (translate.lang.ur.ur class method), 384 capsstart() (translate.lang.vi.vi class method), 385 categories (translate.filters.checks.TeeChecker attribute), 339 categories (translate.filters.checks.UnitChecker attribute), 346 categories (translate.filters.checks.UnitChecker attribute), 3			(translate lang ou ou	class
capsstart() (translate.lang.st.st class method), 379 capsstart() (translate.lang.sv.sv class method), 380 capsstart() (translate.lang.ta.ta class method), 380 capsstart() (translate.lang.te.te class method), 381 capsstart() (translate.lang.th.th class method), 382 capsstart() (translate.lang.ug.ug class method), 382 capsstart() (translate.lang.ur.ur class method), 383 capsstart() (translate.lang.ur.ur class method), 384 capsstart() (translate.lang.vi.vi class method), 384 capsstart() (translate.lang.shi.hi class method), 370 character_iter() (translate.lang.hy.hy class method), 371 character_iter() (translate.lang.km.km class method), 372 character_iter() (translate.lang.kn.kn class method), 373 character_iter() (translate.lang.ko.ko class method), 373 character_iter() (translate.lang.ml.ml class method), 374 character_iter() (translate.lang.ml.ml class method), 375 character_iter() (translate.lang.ml.ml class			(iransiaic.iang.gu.gu	ciuss
capsstart () (translate.lang.sv.sv class method), 380 capsstart () (translate.lang.ta.ta class method), 380 capsstart () (translate.lang.te.te class method), 381 capsstart () (translate.lang.th.th class method), 382 capsstart () (translate.lang.ug.ug class method), 382 capsstart () (translate.lang.ug.ug class method), 383 capsstart () (translate.lang.ug.ug class method), 384 capsstart () (translate.lang.ur.ur class method), 384 capsstart () (translate.lang.vi.vi class method), 385 catchStringOutput (class in translate.lang.vi.vi class method), 385 categories (translate.filters.checks.TeeChecker attribute), 339 categories (translate.filters.checks.UnitChecker attribute), 346 CatkeysDialect (class in translate.storage.catkeys), 414 CatkeysFile (class in translate.storage.catkeys), 415 CatkeysHeader (class in translate.storage.catkeys), 415 CatkeysHeader (class in translate.storage.catkeys), 415 CatkeysHeader (class in translate.storage.catkeys), 6 character_iter () (translate.lang.hi.hi class method), 369 character_iter () (translate.lang.hi.hi class method), 370 character_iter () (translate.lang.hi.hi class method), 371 character_iter () (translate.lang.km.km class method), 373 character_iter () (translate.lang.km.kn class method), 373 character_iter () (translate.lang.ml.ml class method), 374 character_iter () (translate.lang.ml.ml class method), 374 character_iter () (translate.lang.ml.ml class method), 375 character_iter () (translate.lang.ml.ml clas			(translate lang he he	class
capsstart() (translate.lang.ta.ta class method), 380 capsstart() (translate.lang.te.te class method), 381 capsstart() (translate.lang.th.th class method), 382 capsstart() (translate.lang.ug.ug class method), 383 capsstart() (translate.lang.ug.ug class method), 384 capsstart() (translate.lang.ur.ur class method), 384 capsstart() (translate.lang.vi.vi class method), 385 CatchStringOutput (class in translate.misc.wStringIO), 395 categories (translate.filters.checks.TeeChecker attribute), 339 categories (translate.filters.checks.UnitChecker attribute), 346 CatkeysDialect (class in translate.storage.catkeys), 414 CatkeysFile (class in translate.storage.catkeys), 415 CatkeysHeader (class in translate.storage.catkeys), 415	=		(transiate.tang.ne.ne	ciuss
capsstart() (translate.lang.te.te class method), 381 capsstart() (translate.lang.th.th class method), 382 capsstart() (translate.lang.ug.ug class method), 383 capsstart() (translate.lang.ur.ur class method), 384 capsstart() (translate.lang.vi.vi class method), 384 capsstart() (translate.lang.vi.vi class method), 384 capsstart() (translate.lang.zh.zh class method), 385 CatchStringOutput (class in translate.storage.catkeys), 415 categories (translate.filters.checks.UnitChecker attibute), 346 CatkeysFile (class in translate.storage.catkeys), 415 CatkeysHeader (class in translate.storage.catkeys), 415			(translate lang hi hi	class
capsstart() (translate.lang.th.th class method), 382 capsstart() (translate.lang.ug.ug class method), 383 capsstart() (translate.lang.ur.ur class method), 384 capsstart() (translate.lang.ur.ur class method), 384 capsstart() (translate.lang.vi.vi class method), 384 capsstart() (translate.lang.vi.vi class method), 384 capsstart() (translate.lang.xh.zh class method), 385 CatchStringOutput (class in translate.misc.wStringIO), 395 categories (translate.filters.checks.TeeChecker attibute), 339 categories (translate.filters.checks.UnitChecker attibute), 346 CatkeysDialect (class in translate.storage.catkeys), 415 CatkeysHeader (class in translate.storage.catkeys), 415	- · · · · · · · · · · · · · · · · · · ·		(transtate.tang.m.m	ciuss
capsstart() (translate.lang.ug.ug class method), 383 capsstart() (translate.lang.ur.ur class method), 384 capsstart() (translate.lang.vi.vi class method), 384 capsstart() (translate.lang.zh.zh class method), 384 capsstart() (translate.lang.zh.zh class method), 385 CatchStringOutput (class in translate.storage.catkeys), 2414 CatkeysHeader (class in translate.storage.catkeys), 245 CatkeysHeader (class in translate.storag			(translate lang by by	alann
capsstart() (translate.lang.ur.ur class method), 384 capsstart() (translate.lang.vi.vi class method), 384 capsstart() (translate.lang.vi.vi class method), 385 catchStringOutput (class in translate.storage.catkeys), 415 CatkeysHeader (class in translate.storage.catkeys), 415 CatkeysHeader (class in translate.storage.catkeys), 415 CatkeysHeader (class in translate.storage.catkeys), 415 Capsstart() (translate.lang.ur.ur class method), 384 character_iter() (translate.lang.km.km class method), 372 character_iter() (translate.lang.kn.kn class method), 373 character_iter() (translate.lang.ko.ko class method), 373 character_iter() (translate.lang.ml.ml class method), 374 character_iter() (translate.lang.mr.mr class method), 374 character_iter() (translate.lang.mr.mr class method), 375 character_iter() (translate.lang.mr.mr class			(transtate.tang.ny.ny	ciuss
capsstart() (translate.lang.vi.vi class method), 384 capsstart() (translate.lang.zh.zh class method), 385 CatchStringOutput (class in translate.misc.wStringIO), 395 categories (translate.filters.checks.TeeChecker attibute), 339 categories (translate.filters.checks.UnitChecker attibute), 346 CatkeysDialect (class in translate.storage.catkeys), 414 CatkeysFile (class in translate.storage.catkeys), 415 CatkeysHeader (class in translate.storage.catkeys), CatkeysHeader (class in translate.storage.catkeys), 415			(tuanalata lana ia ia	والمعم
capsstart() (translate.lang.zh.zh class method), 385 CatchStringOutput (class in translate.misc.wStringIO), 395 categories (translate.filters.checks.TeeChecker attibute), 339 categories (translate.filters.checks.UnitChecker attibute), 346 CatkeysDialect (class in translate.storage.catkeys), 414 CatkeysFile (class in translate.storage.catkeys), 415 CatkeysHeader (class in translate.storage.catkeys), CatkeysHeader (class in translate.storage.catkeys), 415			(translate.lang.ja.ja	ciass
CatchStringOutput (class in trans- late.misc.wStringIO), 395 categories (translate.filters.checks.TeeChecker at- tribute), 339 categories (translate.filters.checks.UnitChecker at- tribute), 346 CatkeysDialect (class in translate.storage.catkeys), 414 CatkeysFile (class in translate.storage.catkeys), CatkeysHeader (class in trans			(, 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1	1
late.misc.wStringIO), 395 categories (translate.filters.checks.TeeChecker attibute), 339 categories (translate.filters.checks.UnitChecker attibute), 346 CatkeysDialect (class in translate.storage.catkeys), 414 CatkeysFile (class in translate.storage.catkeys), 415 CatkeysHeader (class in translate.storage.catkeys), CatkeysHeader (class in translate.storage.catkeys), 415	- · · · · · · · · · · · · · · · · · · ·		(translate.lang.km.km	class
categories (translate.filters.checks.TeeChecker attribute), 339 categories (translate.filters.checks.UnitChecker attribute), 346 CatkeysDialect (class in translate.storage.catkeys), 414 CatkeysFile (class in translate.storage.catkeys), 415 CatkeysHeader (class in translate.storage.catkeys), 5 CatkeysHeader (class in translate.storage.catkeys), 6 CatkeysHeader (class in translate.storage.catkeys), 7 CatkeysHeader (class in		* *	/. 1 . 1 1 1 1	1
tribute), 339character_iter()(translate.lang.ko.koclasscategories (translate.filters.checks.UnitChecker attribute), 346method), 373(translate.lang.ml.mlclassCatkeysDialect (class in translate.storage.catkeys), 414method), 374(translate.lang.mr.mrclassCatkeysFile (class in translate.storage.catkeys), 415method), 375(translate.lang.mr.mrclassCatkeysHeader (class in translate.storage.catkeys), 415character_iter()(translate.lang.ne.neclass			(translate.lang.kn.kn	class
categories (translate.filters.checks.UnitChecker attribute), 346 character_iter() (translate.lang.ml.ml class  CatkeysDialect (class in translate.storage.catkeys), 414 character_iter() (translate.lang.mr.mr class  CatkeysFile (class in translate.storage.catkeys), 415 method), 375  CatkeysHeader (class in translate.storage.catkeys), character_iter() (translate.lang.ne.ne class				
tribute), 346 character_iter() (translate.lang.ml.ml class method), 374 414 character_iter() (translate.lang.mr.mr class method), 375 CatkeysHeader (class in translate.storage.catkeys), 415 method), 375 CatkeysHeader (class in translate.storage.catkeys), 415 character_iter() (translate.lang.ne.ne class			(translate.lang.ko.ko	class
CatkeysDialect (class in translate.storage.catkeys), 414 character_iter() (translate.lang.mr.mr class CatkeysFile (class in translate.storage.catkeys), 415 method), 375 CatkeysHeader (class in translate.storage.catkeys), character_iter() (translate.lang.ne.ne class				,
414 character_iter() (translate.lang.mr.mr class CatkeysFile(class in translate.storage.catkeys), 415 method), 375 CatkeysHeader (class in translate.storage.catkeys), character_iter() (translate.lang.ne.ne class			(translate.lang.ml.ml	class
CatkeysFile (class in translate.storage.catkeys), 415 method), 375 CatkeysHeader (class in translate.storage.catkeys), character_iter() (translate.lang.ne.ne class				
CatkeysHeader (class in translate.storage.catkeys), character_iter() (translate.lang.ne.ne class			(translate.lang.mr.mr	class
			,	_
416 <i>method</i> ), 376			(translate.lang.ne.ne	class
	416	method), 376		

		,	252	
character_iter()	(translate.lang.pa.pa	class	373	
method), 377		,	characters()	(translate.lang.ko.ko class method),
character_iter()	(translate.lang.si.si	class	373	
method), 378			characters()	(translate.lang.ml.ml class method),
character_iter()	(translate.lang.st.st	class	374	
method), 379		_	characters()	(translate.lang.mr.mr class method),
<pre>character_iter()</pre>	(translate.lang.sv.sv	class	375	
method), 380			characters()	(translate.lang.ne.ne class method),
<pre>character_iter()</pre>	(translate.lang.ta.ta	class	376	
method), 380			characters()	(translate.lang.pa.pa class method),
<pre>character_iter()</pre>	(translate.lang.te.te	class	377	
method), 381				(translate.lang.si.si class method), 378
character_iter()	(translate.lang.th.th	class	characters()	(translate.lang.st.st class method), 379
method), 382			characters()	(translate.lang.sv.sv class method),
<pre>character_iter()</pre>	(translate.lang.ug.ug	class	380	
method), 383			characters()	(translate.lang.ta.ta class method), 380
<pre>character_iter()</pre>	(translate.lang.ur.ur	class	characters()	(translate.lang.te.te class method), 381
method), 384	-		characters()	(translate.lang.th.th class method), 382
character_iter()	(translate.lang.vi.vi	class		(translate.lang.ug.ug class method),
method), 384	·		383	
character_iter()	(translate.lang.zh.zh	class	characters()	(translate.lang.ur.ur class method),
method), 385	, 0		384	
characters() (in mo	odule translate.search.ses	gment).		(translate.lang.vi.vi class method), 384
407		3,		(translate.lang.zh.zh class method),
characters()(transle	ate.lang.af.af.class metho	d). 353	385	(,
characters() (transit	0 0 0		check_values	() (trans-
354	anetten gremment etelss m	,		vert.convert.ArchiveConvertOptionParser
				e neem e name e e neem e promi en se
characters() (trans	slate.lang.ar.ar class m	ethod).	method)	. 233
	slate.lang.ar.ar class m	ethod),	method)	
354	-		check_values	() (trans-
354 characters() (trans	slate.lang.ar.ar class m late.lang.bn.bn class m		check_values late.com	() (trans- vert.convert.ConvertOptionParser
354 characters() (trans	late.lang.bn.bn class m	ethod),	check_values late.com method)	() (transvert.convert.ConvertOptionParser, 237
354 characters() (trans 355 characters() (trans	late.lang.bn.bn class m	ethod),	check_values  late.com method)  check_values	() (trans- vert.convert.ConvertOptionParser , 237 () (trans-
354 characters() (trans 355 characters() (trans method), 356	late.lang.bn.bn class m late.lang.code_or.code_o	ethod), r class	check_values  late.com method) check_values late.com	() (trans- vert.convert.ConvertOptionParser , 237 () (trans- vert.po2moz.MozConvertOptionParser
354 characters() (transisting states) 355 characters() (transisting states) method), 356 characters() (transisting states)	late.lang.bn.bn class m late.lang.code_or.code_o	ethod), r class	check_values	() (transvert.convert.ConvertOptionParser, 237 () (transvert.po2moz.MozConvertOptionParser, 249
354 characters() (transisting transisting to the characters () (transisting transisting transiting transitin	late.lang.bn.bn class m late.lang.code_or.code_o ate.lang.common.Commo	r class	check_values	() (transvert.convert.ConvertOptionParser, 237 () (transvert.po2moz.MozConvertOptionParser, 249 () (transvert.po2moz.MozConvertOptionParser
354 characters() (transisting states () (tran	late.lang.bn.bn class m late.lang.code_or.code_o ate.lang.common.Commo	r class	check_values	() (transvert.convert.ConvertOptionParser, 237 () (transvert.po2moz.MozConvertOptionParser, 249 () (transvert.po2tmx.TmxOptionParser
354 characters() (transismethod), 356 characters() (transismethod), 357 characters() (transismethod), 357 characters() (transismethod), 363	late.lang.bn.bn class m late.lang.code_or.code_o ate.lang.common.Commo late.lang.de.de class m	r class on class eethod),	check_values	() (transvert.convert.ConvertOptionParser, 237 () (transvert.po2moz.MozConvertOptionParser, 249 () (transvert.po2tmx.TmxOptionParser, 254
354 characters() (transismethod), 356 characters() (transismethod), 357 characters() (transismethod), 357 characters() (transismethod), 363 characters() (transismethod)	late.lang.bn.bn class m late.lang.code_or.code_o ate.lang.common.Commo late.lang.de.de class m ate.lang.el.el class metho	nethod), r class on class nethod), d), 364	check_values	() (transvert.convert.ConvertOptionParser, 237 () (transvert.po2moz.MozConvertOptionParser, 249 () (transvert.po2tmx.TmxOptionParser, 254 () (transvert.po2tmx.TmxOptionParser, 254
354 characters() (transismethod), 356 characters() (transismethod), 357 characters() (transismethod), 357 characters() (transismethod), 363 characters() (transismethod), 363	late.lang.bn.bn class m late.lang.code_or.code_o ate.lang.common.Commo late.lang.de.de class m ate.lang.el.el class metho	nethod), r class on class nethod), d), 364	check_values     late.com     method) check_values     late.com     method) check_values     late.com     method) check_values     late.com	() (transvert.convert.ConvertOptionParser, 237 () (transvert.po2moz.MozConvertOptionParser, 249 () (transvert.po2tmx.TmxOptionParser, 254 () (transvert.po2wordfast.WfOptionParser
354 characters() (transismethod), 356 characters() (transismethod), 357 characters() (transismethod), 357 characters() (transismethod), 363 characters() (transismethod), 363	late.lang.bn.bn class m late.lang.code_or.code_o ate.lang.common.Commo late.lang.de.de class m ate.lang.el.el class metho slate.lang.es.es class m	r class on class eethod), d), 364 eethod),	check_values     late.com     method) check_values     late.com     method) check_values     late.com     method) check_values     late.com     method)	() (transvert.convert.ConvertOptionParser, 237 () (transvert.po2moz.MozConvertOptionParser, 249 () (transvert.po2tmx.TmxOptionParser, 254 () (transvert.po2wordfast.WfOptionParser, 259
354 characters() (transisting states of the	late.lang.bn.bn class m late.lang.code_or.code_o ate.lang.common.Commo late.lang.de.de class m ate.lang.el.el class metho clate.lang.es.es class m	r class on class eethod), d), 364 eethod), ed), 366	check_values	() (transvert.convert.ConvertOptionParser, 237 () (transvert.po2moz.MozConvertOptionParser, 249 () (transvert.po2tmx.TmxOptionParser, 254 () (transvert.po2tmx.TmxOptionParser, 259 () (transvert.po2wordfast.WfOptionParser, 259 () (transvert.po2wordfast.WfOptionParser
354 characters() (transismethod), 356 characters() (transismethod), 357 characters() (transismethod), 357 characters() (transismethod) characters() (transismethod) characters() (transismethod) characters() (transismethod) characters() (transismethod)	late.lang.bn.bn class m late.lang.code_or.code_o ate.lang.common.Commo late.lang.de.de class m ate.lang.el.el class metho ate.lang.fa.fa class metho ate.lang.fi.fi class metho ate.lang.fi.fi class metho	r class on class eethod), d), 364 eethod), d), 366 d), 366	check_values     late.com     method) check_values late.filte	() (transvert.convert.ConvertOptionParser, 237 () (transvert.po2moz.MozConvertOptionParser, 249 () (transvert.po2tmx.TmxOptionParser, 254 () (transvert.po2wordfast.WfOptionParser, 259
354 characters() (transismethod), 355 characters() (transismethod), 356 characters() (transismethod), 357 characters() (transismethod) characters() (transismethod) characters() (transismethod) characters() (transismethod) characters() (transismethod) characters() (transismethod)	late.lang.bn.bn class m late.lang.code_or.code_o ate.lang.common.Commo late.lang.de.de class m ate.lang.el.el class metho ate.lang.fa.fa class metho ate.lang.fi.fi class metho ate.lang.fr.fr class metho ate.lang.fr.fr class metho	nethod), r class on class nethod), d), 364 nethod), d), 366 d), 366 d), 367	check_values     late.com     method) check_values     late.com     method) check_values     late.com     method) check_values     late.com     method) check_values     late.filte 348	() (transvert.convert.ConvertOptionParser, 237 () (transvert.po2moz.MozConvertOptionParser, 249 () (transvert.po2tmx.TmxOptionParser, 254 () (transvert.po2vordfast.WfOptionParser, 259 () (transvert.po2wordfast.WfOptionParser, 259 () (transvert.pofilter.FilterOptionParser method),
354 characters() (transismethod), 356 characters() (transismethod), 356 characters() (transismethod), 357 characters() (transismethod)	late.lang.bn.bn class m late.lang.code_or.code_o ate.lang.common.Commo late.lang.de.de class m ate.lang.el.el class metho ate.lang.fa.fa class metho ate.lang.fi.fi class metho ate.lang.fr.fr class metho ate.lang.fr.fr class metho	nethod), r class on class nethod), d), 364 nethod), d), 366 d), 366 d), 367	check_values     late.com     method) check_values     late.com     method) check_values     late.com     method) check_values     late.com     method) check_values     late.filte     348 check_values	() (transvert.convert.ConvertOptionParser, 237 () (transvert.po2moz.MozConvertOptionParser, 249 () (transvert.po2tmx.TmxOptionParser, 254 () (transvert.po2wordfast.WfOptionParser, 259 () (transvert.po2wordfast.WfOptionParser, 259 () (transvert.pofilter.FilterOptionParser method),
characters() (transismethod), 356 characters() (transismethod), 356 characters() (transismethod), 357 characters() (transismethod)	late.lang.bn.bn class m late.lang.code_or.code_o ate.lang.common.Commo late.lang.de.de class m ate.lang.el.el class metho late.lang.fs.fs class metho ate.lang.fi.fi class method ate.lang.fr.fr class method late.lang.gu.gu class m	nethod), or class on class nethod), d), 364 nethod), d), 366 d), 366 d), 367 nethod),	check_values     late.com     method) check_values     late.com     method) check_values     late.com     method) check_values     late.com     method) check_values     late.filte     348 check_values     late.mise	() (transvert.convert.ConvertOptionParser, 237 () (transvert.po2moz.MozConvertOptionParser, 249 () (transvert.po2tmx.TmxOptionParser, 254 () (transvert.po2wordfast.WfOptionParser, 259 () (transvert.po2wordfast.WfOptionParser, 259 () (transvert.pofilter.FilterOptionParser method), () (transvert.pofilter.FilterOptionParser method),
characters() (transismethod), 356 characters() (transismethod), 356 characters() (transismethod), 357 characters() (transismethod), 357 characters() (transismethod)	late.lang.bn.bn class m late.lang.code_or.code_o ate.lang.common.Commo late.lang.de.de class m ate.lang.el.el class metho ate.lang.fa.fa class metho ate.lang.fi.fi class metho ate.lang.fr.fr class metho ate.lang.fr.fr class metho	nethod), or class on class nethod), d), 364 nethod), d), 366 d), 366 d), 367 nethod),	check_values     late.com     method) check_values     late.com     method) check_values     late.com     method) check_values     late.com     method) check_values     late.filte     348 check_values     late.mise     method)	() (transvert.convert.ConvertOptionParser, 237 () (transvert.po2moz.MozConvertOptionParser, 249 () (transvert.po2tmx.TmxOptionParser, 254 () (transvert.po2wordfast.WfOptionParser, 259 () (transvert.po2wordfast.WfOptionParser method), () (transvert.pofilter.FilterOptionParser method), () (transvert.pofilter.FilterOptionParser method), () (transvert.pofilter.FilterOptionParser method),
characters() (transismethod), 356 characters() (transismethod), 356 characters() (transismethod), 357 characters() (transismethod), 357 characters() (transismethod) characters() (transismethod) characters() (transismethod) characters() (transismethod) characters() (transismethod) characters() (transismethod) 368 characters() (transismethod) 368 characters() (transismethod) 369	late.lang.bn.bn class malate.lang.code_or.code_o ate.lang.common.Commo late.lang.de.de class metho ate.lang.el.el class metho ate.lang.fa.fa class metho ate.lang.fi.fi class metho ate.lang.gu.gu class m	r class on class eethod), d), 364 eethod), d), 366 d), 366 d), 367 eethod),	check_values     late.com     method) check_values     late.com     method) check_values     late.com     method) check_values     late.com     method) check_values     late.filte     348 check_values     late.mise     method) check_values	() (transvert.convert.ConvertOptionParser, 237 () (transvert.po2moz.MozConvertOptionParser, 249 () (transvert.po2tmx.TmxOptionParser, 254 () (transvert.po2wordfast.WfOptionParser, 259 () (transvert.po2wordfast.WfOptionParser method), () (transvert.pofilter.FilterOptionParser method), () (transvert.pofilter.FilterOptionParser method), () (transvert.pofilter.FilterOptionParser method), () (transvert.pofilter.FilterOptionParser, 390) () (transvert.pofilter.FilterOptionParser, 390)
characters () (transisters ()	late.lang.bn.bn class m late.lang.code_or.code_o ate.lang.common.Commo late.lang.de.de class m ate.lang.el.el class metho ate.lang.fa.fa class metho ate.lang.fi.fi class metho ate.lang.fi.fi class metho ate.lang.gu.gu class m late.lang.gu.gu class m	nethod), r class on class nethod), d), 364 nethod), d), 366 d), 366 d), 367 nethod), nethod),	check_values     late.com     method) check_values     late.com     method) check_values     late.com     method) check_values     late.filte     348 check_values     late.miss     method) check_values	() (transvert.convert.ConvertOptionParser, 237 () (transvert.po2moz.MozConvertOptionParser, 249 () (transvert.po2tmx.TmxOptionParser, 254 () (transvert.po2tmx.TmxOptionParser, 259 () (transvert.po2wordfast.WfOptionParser, 259 () (transvert.pofilter.FilterOptionParser method), () (transvert.pofilter.FilterOptionParser method), () (transvert.pofilter.FilterOptionParser, 390 () (transvert.pofilter.Security.ConflictOptionParser
characters() (transismethod), 356 characters() (transismethod), 356 characters() (transismethod), 357 characters() (transismethod), 357 characters() (transismethod)	late.lang.bn.bn class m late.lang.code_or.code_o ate.lang.common.Commo late.lang.de.de class m ate.lang.el.el class metho ate.lang.fa.fa class metho ate.lang.fi.fi class metho ate.lang.fi.fi class metho ate.lang.gu.gu class m late.lang.gu.gu class m	nethod), r class on class nethod), d), 364 nethod), d), 366 d), 366 d), 367 nethod), nethod),	check_values     late.com     method) check_values     late.com     method) check_values     late.com     method) check_values     late.filte     348 check_values     late.miss     method) check_values     late.tool method)	() (transvert.convert.ConvertOptionParser, 237 () (transvert.po2moz.MozConvertOptionParser, 249 () (transvert.po2tmx.TmxOptionParser, 254 () (transvert.po2tmx.TmxOptionParser, 259 () (transvert.po2wordfast.WfOptionParser, 259 () (transvert.po2wordfast.WfOptionParser method), () (transvert.pofilter.FilterOptionParser method), () (transvert.pofilter.FilterOptionParser method), () (transvert.pofilter.FilterOptionParser, 390 () (transvert.poconflicts.ConflictOptionParser, 683
characters() (transismethod), 356 characters() (transismethod), 356 characters() (transismethod), 357 characters() (transismethod), 357 characters() (transismethod)	late.lang.bn.bn class malate.lang.code_or.code_o ate.lang.common.Commo late.lang.de.de class m ate.lang.el.el class metho ate.lang.fs.fa class metho ate.lang.fr.fr class metho ate.lang.gu.gu class m late.lang.he.he class m ate.lang.he.he class metho ate.lang.he.he class m	nethod), or class on class nethod), od), 364 nethod), od), 366 d), 367 nethod), nethod), od), 369 nethod),	check_values     late.com     method) check_values     late.com     method) check_values     late.com     method) check_values     late.filte     348 check_values     late.mise     method) check_values     late.tool     method) check_values	() (transvert.convert.ConvertOptionParser, 237 () (transvert.po2moz.MozConvertOptionParser, 249 () (transvert.po2tmx.TmxOptionParser, 254 () (transvert.po2tmx.TmxOptionParser, 259 () (transvert.po2wordfast.WfOptionParser, 259 () (transvert.pofilter.FilterOptionParser method), () (transvert.pofilter.FilterOptionParser method), () (transvert.pofilter.FilterOptionParser, 390 () (transvert.poconflicts.ConflictOptionParser, 683 () (transvert.poconflicts.ConflictOptionParser, 683 () (transvert.poconflicts.ConflictOptionParser, 683 () (transvert.poconflicts.ConflictOptionParser, 683
characters () (transismethod), 356 characters () (transismethod), 356 characters () (transismethod), 357 characters () (transismethod), 357 characters () (transismethod)	late.lang.bn.bn class m late.lang.code_or.code_o ate.lang.common.Commo late.lang.de.de class m ate.lang.el.el class metho ate.lang.fa.fa class metho ate.lang.fi.fi class metho ate.lang.fr.fr class metho late.lang.gu.gu class m late.lang.he.he class m ate.lang.hi.hi class metho ate.lang.hi.hi class metho ate.lang.hi.hy class m	nethod), or class on class nethod), d), 364 nethod), d), 366 d), 367 nethod), nethod), d), 369 nethod),	check_values     late.com     method) check_values     late.com     method) check_values     late.com     method) check_values     late.filte     348 check_values     late.miss     method) check_values     late.tool     method) check_values	() (transvert.convert.ConvertOptionParser, 237 () (transvert.po2moz.MozConvertOptionParser, 249 () (transvert.po2tmx.TmxOptionParser, 254 () (transvert.po2tmx.TmxOptionParser, 259 () (transvert.po2wordfast.WfOptionParser, 259 () (transvert.po2wordfast.WfOptionParser method), () (transvert.pofilter.FilterOptionParser method), () (transvert.pofilter.FilterOptionParser method), () (transvert.pofilter.FilterOptionParser, 390 () (transvert.poconflicts.ConflictOptionParser, 683
characters () (translements)	late.lang.bn.bn class m late.lang.code_or.code_o ate.lang.common.Commo late.lang.de.de class m ate.lang.el.el class metho ate.lang.fa.fa class metho ate.lang.fi.fi class metho ate.lang.fr.fr class metho late.lang.gu.gu class m late.lang.he.he class m ate.lang.hi.hi class metho ate.lang.hi.hi class metho ate.lang.hi.hy class m	nethod), or class on class nethod), d), 364 nethod), d), 366 d), 367 nethod), nethod), d), 369 nethod),	check_values late.com method) check_values late.com method) check_values late.com method) check_values late.filte 348 check_values late.miss method) check_values late.tool method) check_values	() (transvert.convert.ConvertOptionParser, 237 () (transvert.po2moz.MozConvertOptionParser, 249 () (transvert.po2tmx.TmxOptionParser, 254 () (transvert.po2wordfast.WfOptionParser, 259 () (transvert.po2wordfast.WfOptionParser method), () (transvert.pofilter.FilterOptionParser method), () (transvert.pofilter.FilterOptionParser method), () (transvert.pofilter.FilterOptionParser method), () (transvert.pofilter.FilterOptionParser, 390 () (transvert.pofilter.ConflictOptionParser, 683 () (transvert.pofilter.ConflictOptionParser method),
characters() (translemethod), 356 characters() (translemethod), 356 characters() (translemethod), 357 characters() (translemethod), 357 characters() (translemethod), 357 characters() (translemethod), 363 characters() (translemethod), 365 characters() (translemethod), 365 characters() (translemethod), 368 characters() (translemethod), 368 characters() (translemethod), 369 characters() (translemethod), 370 characters() (translemethod), 370 characters() (translemethod), 372	late.lang.bn.bn class made.lang.code_or.code_o ate.lang.common.Commo late.lang.de.de class m ate.lang.el.el class metho ate.lang.fa.fa class metho ate.lang.fr.fr class metho ate.lang.gu.gu class m late.lang.he.he class m ate.lang.hy.hy class m ate.lang.hy.hy class m ate.lang.ja.ja class metho ate.lang.hy.hy class m	nethod), r class on class nethod), d), 364 nethod), d), 366 d), 367 nethod), nethod), d), 369 nethod), d), 371 nethod),	check_values late.com method) check_values late.com method) check_values late.com method) check_values late.com method) check_values late.filte 348 check_values late.mise method) check_values late.tool method) check_values late.tool fe87 check_values	() (transvert.convert.ConvertOptionParser, 237 () (transvert.po2moz.MozConvertOptionParser, 249 () (transvert.po2tmx.TmxOptionParser, 254 () (transvert.po2wordfast.WfOptionParser, 259 () (transvert.po2wordfast.WfOptionParser method), () (transvert.pofilter.FilterOptionParser method), () (transvert.pofilter.FilterOptionParser method), () (transvert.pofilter.FilterOptionParser, 390 () (transvert.pofilter.ConflictOptionParser, 683 () (transvert.pofilter.ConflictOptionParser method), () (transvert.pofilter.ConflictOptionParser method),
characters() (translemethod), 356 characters() (translemethod), 356 characters() (translemethod), 357 characters() (translemethod), 357 characters() (translemethod), 357 characters() (translemethod), 363 characters() (translemethod), 365 characters() (translemethod), 365 characters() (translemethod), 368 characters() (translemethod), 368 characters() (translemethod), 369 characters() (translemethod), 370 characters() (translemethod), 370 characters() (translemethod), 372	late.lang.bn.bn class m late.lang.code_or.code_o ate.lang.common.Commo late.lang.de.de class m ate.lang.el.el class metho ate.lang.fa.fa class metho ate.lang.fi.fi class metho ate.lang.fr.fr class metho late.lang.gu.gu class m late.lang.he.he class m ate.lang.hi.hi class metho ate.lang.hi.hi class metho ate.lang.hi.hy class m	nethod), r class on class nethod), d), 364 nethod), d), 366 d), 367 nethod), nethod), d), 369 nethod), d), 371 nethod),	check_values late.com method) check_values late.com method) check_values late.com method) check_values late.com method) check_values late.filte 348 check_values late.mise method) check_values late.tool method) check_values late.tool fe87 check_values	() (transvert.convert.ConvertOptionParser, 237 () (transvert.po2moz.MozConvertOptionParser, 249 () (transvert.po2tmx.TmxOptionParser, 254 () (transvert.po2wordfast.WfOptionParser, 259 () (transvert.po2wordfast.WfOptionParser method), () (transvert.pofilter.FilterOptionParser method), () (transvert.pofilter.FilterOptionParser method), () (transvert.pofilter.FilterOptionParser method), () (transvert.pofilter.FilterOptionParser, 390 () (transvert.pofilter.ConflictOptionParser, 683 () (transvert.pofilter.ConflictOptionParser method),

method), 690		late.convert.convert.ConvertOptionParser
_	rans-	method), 237
late.tools.poterminology.TerminologyOptic	onParser:he	eckoutputsubdir() (trans-
method), 694		late.convert.po2moz.MozConvertOptionParser
checker (translate.lang.common.Common attrib	oute),	method), 249
357	che	eckoutputsubdir() (trans-
checker_name (ta	rans-	late.convert.po2tmx.TmxOptionParser
late. filters. checks. CCLicense Checker	at-	method), 254
tribute), 269	che	eckoutputsubdir() (trans-
	rans-	late.convert.po2wordfast.WfOptionParser
late.filters.checks.DrupalChecker attrib	oute),	method), 259
275	che	eckoutputsubdir() (trans-
checker_name (to	rans-	$late.filters.pofilter.FilterOptionParser\ method),$
late.filters.checks.GnomeChecker attrib	oute),	348
281	che	eckoutputsubdir() (trans-
checker_name (translate.filters.checks.IOSCheattribute), 287	ecker	late.misc.optrecurse.RecursiveOptionParser method), 390
checker_name (translate.filters.checks.KdeChe	ecker che	eckoutputsubdir() (trans-
attribute), 292		late.tools.poconflicts.ConflictOptionParser
checker_name (translate.filters.checks.L20nChe	ecker	method), 683
attribute), 298		eckoutputsubdir() (trans-
checker_name (ta	rans-	late.tools.pogrep.GrepOptionParser method),
late.filters.checks.LibreOfficeChecker	at-	687
tribute), 304	che	eckoutputsubdir() (trans-
checker_name (ta	rans-	late.tools.porestructure.SplitOptionParser
late.filters.checks.MinimalChecker attrib	oute),	method), 690
310	che	eckoutputsubdir() (trans-
checker_name (ta	rans-	late.tools.poterminology.TerminologyOptionParser
late.filters.checks.MozillaChecker attrib	oute),	method), 694
315		dict (class in translate.misc.dictutils), 386
checker_name (ta	rans- CJI	Kpunc (translate.lang.common.Common attribute),
late.filters.checks.OpenOfficeChecker	at-	357
tribute), 321	cla	assifyunit() (trans-
checker_name (ta	rans-	late.storage.statistics.Statistics method),
late.filters.checks.ReducedChecker attrib	oute),	610
327	cla	assifyunits() (trans-
checker_name (ta	rans-	late.storage.statistics.Statistics method),
late.filters.checks.StandardChecker attrib	oute),	610
333	clo	dr_plural_categories (in module trans-
checker_name (ta	rans-	late.lang.data), 360
late.filters.checks.StandardUnitChecker tribute), 338	at- cle	ean () (translate.tools.poconflicts.ConflictOptionParser method), 683
checker_name (translate.filters.checks.TermChe	ecker cle	eanfile() (in module translate.tools.poclean), 682
attribute), 340		eanunit () (in module translate.tools.poclean), 682
checker_name (ta		eanup()(translate.storage.bundleprojstore.BundleProjectSto
late.filters.checks.TranslationChecker	at-	method), 413
tribute), 345	cle	ear () (translate.misc.dictutils.cidict method), 386
checker_name (translate.filters.checks.UnitChe	ecker cle	ear() (translate.storage.oo.unormalizechar
attribute), 346		method), 496
CheckerConfig (class in translate.filters.checks),	,274 cle	ear_test_dir() (trans-
	rans-	late.storage.benchmark.TranslateBenchmarker
late.convert.convert.ArchiveConvertOption	Parser	method), 413
method), 234		ose() (translate.misc.wStringIO.CatchStringOutput
checkoutputsubdir() (ta	rans-	method), 395

close() (translate.storage.html.htmlfile method), 435	275
close() (translate.storage.html.POHTMLParser	compendiumconflicts() (trans-
method), 432	late.filters.checks.GnomeChecker method),
<pre>close() (translate.storage.project.Project method),</pre>	281
567	compendiumconflicts() (trans-
code (translate.lang.common.Common attribute), 357	late.filters.checks.IOSChecker method), 287
code_or (class in translate.lang.code_or), 356	compendiumconflicts() (trans-
combine() (in module translate.convert.accesskey), 232	late.filters.checks.KdeChecker method), 293
combine() (translate.storage.catkeys.CatkeysDialect class method), 414	compendiumconflicts() (trans- late.filters.checks.L2OnChecker method),
combine() (translate.storage.omegat.OmegaTDialect	298
class method), 487	compendiumconflicts() (trans-
combine() (translate.storage.utx.UtxDialect class	late.filters.checks.LibreOfficeChecker method),
method), 656	304
<pre>combine() (translate.storage.wordfast.WordfastDialect</pre>	compendiumconflicts() (trans-
class method), 666	late.filters.checks.MinimalChecker method),
commit() (translate.storage.versioncontrol.bzr.bzr	310
method), 663	compendiumconflicts() (trans-
commit() (translate.storage.versioncontrol.cvs.cvs method), 663	late.filters.checks.MozillaChecker method), 315
commit () (translate.storage.versioncontrol.darcs.darcs	compendiumconflicts() (trans-
method), 664	late.filters.checks.OpenOfficeChecker method),
commit () (translate.storage.versioncontrol.GenericRevis	
method), 662	compendiumconflicts() (trans-
commit() (translate.storage.versioncontrol.git.git	late.filters.checks.ReducedChecker method),
method), 664	327
commit() (translate.storage.versioncontrol.hg.hg	compendiumconflicts() (trans-
method), 665	late.filters.checks.StandardChecker method),
commit() (translate.storage.versioncontrol.svn.svn	333
method), 665	compendiumconflicts() (trans-
commit_transaction() (trans-	late.filters.checks.TermChecker method),
late.search.indexing.CommonIndexer.CommonDe	
method), 397	compose_mappings() (in module trans-
commit_transaction() (trans-	late.storage.xml_extract.misc), 680
_	Datahásteanslate.storage.statsdb.StatsCache attribute), 611
method), 400	ConflictOptionParser (class in trans-
	late.tools.poconflicts), 683
	pasensoleColor (class in translate.tools.pocount), 686
method), 403	convert_forward() (trans-
commitdirectory() (in module trans-	late.storage.project.Project method), 567
late.storage.versioncontrol), 662	convert_store() (in module trans-
Common (class in translate.lang.common), 357	late.convert.sub2po), 265
CommonDatabase (class in trans-	convert_store() (translate.convert.ical2po.ical2po
late.search.indexing.CommonIndexer), 397	method), 242
CommonEnquire (class in trans-	convert_store() (translate.convert.ini2po.ini2po
late.search.indexing.CommonIndexer), 399	method), 243
commonpunc (translate.lang.common.Common at-	
tribute), 358	late.convert.json2po.json2po method), 243
compendiumconflicts() (trans-	convert_store() (trans-
late.filters.checks.CCLicenseChecker method), 269	late.convert.mozlang2po.lang2po method), 244
compendiumconflicts() (trans-	convert_store() (translate.convert.php2po.php2po
late.filters.checks.DrupalChecker method),	method), 246

convert_store() (trans-	convertcsv() (in module translate.convert.po2csv).
late.convert.po2mozlang.po2lang method),	246
248	convertdtd() (in module translate.convert.dtd2po).
<pre>convert_store() (translate.convert.po2tiki.po2tiki</pre>	241
method), 254	convertfile() (translate.convert.csv2tbx.csv2tbx
<pre>convert_store() (translate.convert.po2txt.po2txt</pre>	method), 241
method), 258	converthtml() (in module trans-
<pre>convert_store() (translate.convert.rc2po.rc2po</pre>	late.convert.html2po), 242
method), 265	converthtml() (in module trans-
convert_store() (trans-	late.convert.po2html), 247
late.convert.resx2po.resx2po method), 265	convert json () (in module trans-
convert_store() (translate.convert.tiki2po.tiki2po	late.convert.json2po), 243
method), 266	convertmo() (in module translate.tools.pocompile).
	682
convert_store() (translate.convert.txt2po.txt2po	
method), 267	convertmozillaprop() (in module trans-
convert_store() (trans-	late.convert.po2prop), 252
late.convert.yaml2po.yaml2po method),	convertmozillaprop() (in module trans-
268	late.convert.prop2po), 264
convert_stores() (in module trans-	convertodf() (in module translate.convert.odf2xliff).
late.convert.pot2po), 263	245
convert_unit() (in module trans-	convertoo() (in module translate.convert.oo2po)
late.convert.sub2po), 265	245
<pre>convert_unit() (translate.convert.ical2po.ical2po</pre>	convertoo() (in module translate.convert.oo2xliff).
method), 242	245
<pre>convert_unit() (translate.convert.ini2po.ini2po</pre>	ConvertOptionParser (class in trans-
method), 243	late.convert.convert), 237
convert_unit() (translate.convert.json2po.json2po	convertphp2py() (in module trans-
method), 243	late.tools.phppo2pypo), 682
convert_unit() (trans-	convertpo() (in module translate.convert.po2tmx).
late.convert.mozlang2po.lang2po method),	257
244 method),	
	convertpo() (in module translate.convert.po2ts), 258
convert_unit() (translate.convert.php2po.php2po	convertpo() (in module trans-
method), 246	late.convert.po2wordfast), 262
convert_unit() (trans-	convertpo() (in module translate.convert.po2xliff).
late.convert.po2mozlang.po2lang method),	262
248	convertpo() (in module translate.tools.podebug), 686
convert_unit() (translate.convert.po2tiki.po2tiki	convertpo() (in module translate.tools.poswap), 693
method), 254	convertpot() (in module translate.convert.pot2po)
convert_unit() (trans-	263
late.convert.po2yaml.po2yaml method),	convertprop() (in module trans-
263	late.convert.prop2po), 264
<pre>convert_unit() (translate.convert.rc2po.rc2po</pre>	convertpropunit() (trans-
method), 265	late.convert.prop2po.prop2po method), 264
<pre>convert_unit() (translate.convert.resx2po.resx2po</pre>	convertpy2php() (in module trans-
method), 265	late.tools.pypo2phppo), 698
<pre>convert_unit() (translate.convert.tiki2po.tiki2po</pre>	convertrc() (in module translate.convert.rc2po), 264
method), 266	convertstore() (translate.convert.csv2po.csv2po
convert_unit() (trans-	method), 241
late.convert.yaml2po.yaml2po method),	convertstore() (translate.convert.prop2po.prop2po
268	method), 264
CONTRACTORIO LIN MODULO TRANSLATO CONVORT CEN /NO.	
convertcsv() (in module translate.convert.csv2po),	convertstrings() (in module trans-
240	late.convert.po2prop), 253

convertsub() (in module translate.convert.sub2po), 265	copy () (translate.storage.placeables.xliff.Bx method), 535
convertts () (in module translate.convert.ts2po), 266	copy() (translate.storage.placeables.xliff.Ept method),
convertunit() (translate.convert.csv2po.csv2po	532
method), 241	copy() (translate.storage.placeables.xliff.Ex method),
convertunit() (translate.convert.prop2po.prop2po	537
method), 264	copy() (translate.storage.placeables.xliff.G method).
convertxliff() (in module trans-	538
late.convert.xliff2odf), 267	copy() (translate.storage.placeables.xliff.It method).
convertxliff() (in module trans-	540
late.convert.xliff2po), 268	copy() (translate.storage.placeables.xliff.Ph method),
copy () (translate.misc.dictutils.cidict method), 386	543
copy() (translate.storage.oo.unormalizechar method),	copy () (translate.storage.placeables.xliff.Sub method),
496	541
copy() (translate.storage.placeables.base.Bpt method),	copy () (translate.storage.placeables.xliff.UnknownXMI
497	method), 545
copy() (translate.storage.placeables.base.Bx method),	copy() (translate.storage.placeables.xliff.X method),
505	533
copy() (translate.storage.placeables.base.Ept method),	copy () (translate.storage.tmx.tmxunit method), 637
499	copyinput() (in module translate.convert.convert),
copy() (translate.storage.placeables.base.Ex method),	240
507	copytemplate() (in module trans-
copy() (translate.storage.placeables.base.G method),	late.convert.convert), 240
504	correct() (in module translate.filters.autocorrect),
copy() (translate.storage.placeables.base.It method), 502	268
	correctorigin() (trans-
copy () (translate.storage.placeables.base.Ph method), 500	late.storage.poxliff.PoXliffUnit method), 564
copy () (translate.storage.placeables.base.Sub method),	correctorigin() (translate.storage.xliff.xliffunit
510	method), 675
copy() (translate.storage.placeables.base.X method),	count() (translate.misc.multistring.multistring
508	method), 387
copy () (translate.storage.placeables.general.AltAttrPlace	
method), 512	late.filters.decoration), 347
copy () (translate.storage.placeables.general.XMLEntityF	· ·
method), 514	348
copy () (translate.storage.placeables.general.XMLTagPla	co <b>eobh</b> etsmatch() (in module translate.filters.helpers),
method), 515	348
copy () (translate.storage.placeables.interfaces.BasePlace	$em{able}$ ntwords () (translate.storage.statistics.Statistics
method), 517	method), 611
copy () (translate.storage.placeables.interfaces.InvisibleF	
method), 519	late. storage. benchmark. Translate Benchmarker
copy () (translate.storage.placeables.interfaces.MaskingF	
method), 520	createcontextgroup() (trans-
copy () (translate.storage.placeables.interfaces.Replacem	
method), 522	564
copy () (translate.storage.placeables.interfaces.SubflowP	
method), 524	late.storage.xliff.xliffunit method), 676
copy () (translate.storage.placeables.strelem.StringElem	
method), 526	late.storage.poxliff.PoXliffFile method), 560
copy () (translate.storage.placeables.terminology.Termino method), 528	ology Blaedable node () (translate.storage.xliff.xlifffile method), 673
copy () (translate.storage.placeables.xliff.Bpt method),	
530	method) 560

creategroup()	(translate. storage. xliff. xlifffile	D	
method), 673		darcs (class in translate.storage.versioncontrol.	.darcs),
createlanguageNod		664	
	a.LISAunit method), 473	de (class in translate.lang.de), 363	
createlanguageNod		decode() (translate.misc.multistring.mul	tistring
	xliff.PoXliffUnit method),	method), 387	
564		decode() (translate.storage.pypo.pofile method)	), 589
createlanguageNod		decode_header() (in module	trans-
- I	h.QphUnit method), 603	late.storage.poparser), 559	
createlanguageNod		DEFAULT_RCS (in module	trans-
	x.tbxunit method), 626	late.storage.versioncontrol), 661	
createlanguageNod		<pre>define_option()</pre>	(trans-
	x.tmxunit method), 637	late.convert.convert.ArchiveConvertOpti	ionParser
createlanguageNod		method), 234	
	Lisunit method), 648	<pre>define_option()</pre>	(trans-
createlanguageNod		late.convert.convert.ConvertOptionParse	er
	ff.xliffunit method), 676	method), 237	
createsubfileinde		<pre>define_option()</pre>	(trans-
	.oomultifile method), 495 filters.checks.CCLicenseChecker	late.convert.po2moz.MozConvertOption.	Parser
method), 269	miers.checks.ccLicenseChecker	method), 249	
	ata filtars abacks DrunalChacker	<pre>define_option()</pre>	(trans-
credits() (translemethod), 275	ate.filters.checks.DrupalChecker	late.convert.po2tmx.TmxOptionParser	
* *	ate.filters.checks.GnomeChecker	method), 254	
method), 281	are.jurers.checks.OnomeChecker	<pre>define_option()</pre>	(trans-
	nslate.filters.checks.IOSChecker	late.convert.po2wordfast.WfOptionParse	er
method), 287	msiate.juters.checks.105Checker	method), 259	
/ /	nslate.filters.checks.KdeChecker	define_option()	(trans-
method), 293	instanci, tine i si cine cius. Hac e incene i	late.filters.pofilter.FilterOptionParser m	etnoa),
* *	slate.filters.checks.L20nChecker	348	(4
method), 298	and the state of t	define_option()	(trans-
	filters.checks.LibreOfficeChecker	late.misc.optrecurse.RecursiveOptionPa.	rser
method), 304		<pre>method), 390 define_option()</pre>	(trans-
	te.filters.checks.MinimalChecker	late.tools.poconflicts.ConflictOptionPars	`
method), 310	,	method), 683	SE1
	ate.filters.checks.MozillaChecker	define_option()	(trans-
method), 316			ethod),
credits()(translate.f	filters.checks.OpenOfficeChecker	687	cinou),
method), 321		define_option()	(trans-
credits() (translat	te.filters.checks.ReducedChecker	late.tools.porestructure.SplitOptionParso	`
method), 327		method), 690	
credits() (translate	e.filters.checks.StandardChecker	define_option()	(trans-
method), 333		late.tools.poterminology.TerminologyOp	*
	slate.filters.checks.TermChecker	method), 694	
method), 340		delalttrans() (translate.storage.poxliff.PoX	liffUnit
csv2po (class in transla	ate.convert.csv2po), 241	method), 564	- 93
csv2tbx (class in trans	elate.convert.csv2tbx), 241	delalttrans() (translate.storage.xliff.x	xliffunit
csvfile (class in trans	slate.storage.csvl10n), 420	method), 676	
	slate.storage.csvl10n), 421	delete_doc()	(trans-
	c.lru.LRUCachingDict method),	late.search.indexing.CommonIndexer.Co	mmonDatabase
387		method), 397	
_	atsdb.StatsCache attribute), 611	delete_doc()	(trans-
	torage.versioncontrol.cvs), 663	late.search.indexing.PyLuceneIndexer.P	yLuceneDatabas
cyr21at (in module tra	inslate.lang.af), 353	method), 400	

```
delete doc()
                                              (trans- delete range()
                                                                                                     (trans-
         late.search.indexing.XapianIndexer.XapianDatabase
                                                                late.storage.placeables.interfaces.MaskingPlaceable
        method), 403
                                                                method), 521
delete_document_by_id()
                                              (trans- delete_range()
                                                                                                     (trans-
         late.search.indexing.CommonIndexer.CommonDatabase
                                                                late. storage. placeables. interfaces. Replacement Placeable\\
        method), 398
                                                                method), 522
delete document by id()
                                              (trans- delete range()
                                                                                                     (trans-
         late.search.indexing.PyLuceneIndexer.PyLuceneDatabase late.storage.placeables.interfaces.SubflowPlaceable
        method), 400
                                                                method), 524
                                              (trans- delete_range()
                                                                                                     (trans-
delete_document_by_id()
         late.search.indexing.XapianIndexer.XapianDatabase
                                                                late. storage. placeables. strelem. String Elem\\
        method), 403
                                                                method), 526
                                                      delete_range()
delete_range()
                                              (trans-
                                                                                                     (trans-
         late.storage.placeables.base.Bpt
                                                                late.storage.placeables.terminology.TerminologyPlaceable
                                            method),
         497
                                                                method), 528
delete_range()
                                              (trans-
                                                      delete_range()
                                                                                                     (trans-
         late.storage.placeables.base.Bx
                                                                late.storage.placeables.xliff.Bpt
                                            method),
                                                                                                   method),
         505
                                                                530
                                                                                                     (trans-
delete range()
                                              (trans-
                                                      delete range()
         late.storage.placeables.base.Ept
                                            method),
                                                                late.storage.placeables.xliff.Bx
                                                                                                   method),
         499
                                                                535
delete_range()
                                              (trans-
                                                      delete range()
                                                                                                     (trans-
         late.storage.placeables.base.Ex
                                                                late.storage.placeables.xliff.Ept
                                                                                                   method),
                                            method),
         507
                                                                532
delete_range()
                                              (trans-
                                                      delete range()
                                                                                                     (trans-
         late.storage.placeables.base.G
                                            method),
                                                                late.storage.placeables.xliff.Ex
                                                                                                   method),
                                                      {\tt delete\_range} () (translate.storage.placeables.xliff.G
delete_range()
                                              (trans-
         late.storage.placeables.base.It
                                                                method), 538
                                            method),
         502
                                                       delete_range() (translate.storage.placeables.xliff.It
delete_range()
                                              (trans-
                                                                method), 540
         late.storage.placeables.base.Ph
                                            method),
                                                      delete_range()
                                                                                                     (trans-
         500
                                                                late.storage.placeables.xliff.Ph
                                                                                                   method),
delete_range()
                                              (trans-
         late.storage.placeables.base.Sub
                                            method),
                                                      delete range()
                                                                                                     (trans-
                                                                late.storage.placeables.xliff.Sub
                                                                                                   method),
delete range()
                                              (trans-
                                                                542
         late.storage.placeables.base.X
                                            method),
                                                      delete_range()
                                                                                                     (trans-
         509
                                                                late.storage.placeables.xliff.UnknownXML
                                                                method), 545
delete_range()
                                              (trans-
         late.storage.placeables.general.AltAttrPlaceable delete range() (translate.storage.placeables.xliff.X
        method), 512
                                                                method), 533
delete range()
                                              (trans- depth first()
                                                                                                     (trans-
         late.storage.placeables.general.XMLEntityPlaceable
                                                                late.storage.placeables.base.Bpt
                                                                                                   method),
        method), 514
delete_range()
                                              (trans- depth_first() (translate.storage.placeables.base.Bx
         late.storage.placeables.general.XMLTagPlaceable
                                                                method), 505
        method), 515
                                                       depth_first()
                                                                                                     (trans-
delete_range()
                                              (trans-
                                                                late.storage.placeables.base.Ept
                                                                                                   method),
         late.storage.placeables.interfaces.BasePlaceable
        method), 517
                                                      depth_first() (translate.storage.placeables.base.Ex
delete_range()
                                              (trans-
                                                                method), 507
         late.storage.placeables.interfaces.InvisiblePlaceable pth\_first() (translate.storage.placeables.base.G
                                                                method), 504
        method), 519
```

depth_	_first() (translate.storage.placeable method), 502	es.base.It	depth_:	first() (trans method), 543	slate.storage.placeal	ples.xliff.Ph
depth_	_first()	(trans-	depth_:	first()		(trans-
	late.storage.placeables.base.Ph 501	method),		late.storage.plac 542	ceables.xliff.Sub	method),
depth_	_first()	(trans-	depth_:	first()		(trans-
		method),	_		ceables.xliff.Unknow	nXML
depth_	_first() (translate.storage.placeable method), 509	es.base.X	depth_:	first() (tran method), 534	nslate.storage.placed	ıbles.xliff.X
depth_	_first()	(trans-	destro	y () (translate.co	onvert.convert.Archiv	veConvertOptionParser
	late.storage.placeables.general.AltAttr	<i>Placeable</i>		method), 234		
	method), 512		destro	y () (translate.co	onvert.convert.Conve	rtOptionParser
depth_	_first()	(trans-		method), 237		
	late.storage.placeables.general.XMLE method), 514	IntityPlaced	a <b>ble</b> stro <u>r</u>	y () (translate.co method), 249	onvert.po2moz.MozC	ConvertOptionParser
depth_	_first()			y () (translate.co	onvert.po2tmx.TmxO	ptionParser
	late.storage.placeables.general.XMLT	agPlaceabl	le	method), 254		
	method), 516		destro	y () (translate.co	onvert.po2wordfast.V	VfOptionParser
depth_	_first()	(trans-		method), 259		
	late.storage.placeables.interfaces.Base method), 517	ePlaceable	destro	y () (translate.fii method), 348	lters.pofilter.FilterOp	otionParser
depth_	_first()	(trans-	destro	y () (translate.m	isc.optrecurse.Recur	siveOptionParser
	late.storage.placeables.interfaces.Invis	siblePlaced	ıble	method), 390		
	method), 519		destro	y () (translate.to	ols.poconflicts.Confl	lictOptionParser
depth_	_first()	(trans-		method), 683		
	late.storage.placeables.interfaces.Mas	kingPlaced	<i>ıldle</i> stro		tools.pogrep.GrepO <sub>l</sub>	otionParser
	method), 521			method), 687		
depth_	_first()				ols.porestructure.Sp	litOptionParser
	late.storage.placeables.interfaces.Rep	lacementPl				
	method), 522		destro		ols.poterminology.Te	erminologyOptionPars
depth_	_first()	(trans-		method), 694		
	late.storage.placeables.interfaces.Subj	flowPlacea	$\mathit{bb}\!$			(trans-
	method), 524				e.TranslationStore	method),
depth_	_first()	(trans-	_	408		,
	late.storage.placeables.strelem.Stringl	Elem	detect_	_encoding()	G 1 F1	(trans-
	method), 526			late.storage.catk	xeys.CatkeysFile	method),
depth_	_first()	(trans-	Dh . 11.	415		(1
	late.storage.placeables.terminology.Te	rminology	Pate elde		110	(trans-
al a -a ± la	method), 528	(tuana	al a ± a a ±		10n.csvfile method),	
aeptn_	_first()	(trans-	detect_	_encoding()	(translate.storage	г.ата.атајне
	late.storage.placeables.xliff.Bpt 530	method),	1 - 4 4	method), 426	(4	41 1.41C1 -
don+h		ag vliff Dv	detect_		(translate.storage.h	іті.пітіјне
deptii_	_first() (translate.storage.placeable method), 535	es.xujj.bx	do+ o	method), 435 _encoding()		(trans-
donth	_first()	(trans-	detect_	_	l.POHTMLParser	method),
deptii_		method),		432	i.i OiiimLi arser	meinoa),
	532		detect_	_encoding()	(translate.storage.	ical.icalfile
aepth_	_first() (translate.storage.placeable	es.xuŋf.Ex	3 - 4 - 1	method), 441	(4	:: ::::::::::::::::::::::::::::
1 1 3	method), 537	l 1:0° C	aetect_	_encoding()	(translate.storag	ge.ini.inifile
aepth_	_first() (translate.storage.placeable	es.xiiff.G	d a + a a +	method), 446		(tuana
don+h	method), 538	les vliff It	uerect_	_encoding()	110n I18NovtEile	(trans- method),
aehrii_	_first() (translate.storage.placeab method), 540	ıes.x11]J.11		452	ıl10n.I18NextFile	тетоа),

detect	_encoding() (translate.storage.jsonl10n.JsonFile method), 457	- late.storage.subtitles.AdvSubStationAlphaFile method), 613
detect_	_encoding() (translate.storage.jsonl10n.JsonNestedFile method 458	
detect_	_encoding() (translate.storage.jsonl10n.WebExtensionJsonFile method), 466	- detect_encoding() (trans- late.storage.subtitles.SubRipFile method), 616
detect_	_encoding() (translate.storage.lisa.LISAfil method), 471	e detect_encoding() (trans- late.storage.subtitles.SubStationAlphaFile
detect_	_encoding() (translate.storage.mo.mofilemethod), 477	e method), 618 detect_encoding() (trans-
detect_	_encoding() (translate.storage.mozilla_lang.LangStore method	, 619
detect_	482 _encoding() (trans	· · · · · · · · · · · · · · · · · · ·
	late.storage.omegat.OmegaTFile method 488	, detect_encoding() (translate.storage.tiki.TikiStore method), 630
detect_	_encoding() (trans late.storage.omegat.OmegaTFileTab method	_
detect_	489 _encoding() (translate.storage.php.phpfil method), 547	<pre>detect_encoding() (trans-</pre>
detect_	_encoding() (trans	- detect_encoding() (translate.storage.ts2.tsfile
detect	late.storage.pocommon.pofile method), 552 _encoding() (translate.storage.poxliff.PoXliffFile method), 560	<pre>method), 645 - detect_encoding() (translate.storage.txt.TxtFile     method), 651</pre>
detect_	_encoding() (translate.storage.properties.javafile method	- detect_encoding() (translate.storage.utx.UtxFile
detect <sub>.</sub>	575 _encoding() (translate.storage.properties.javautf8file method	, 667
detect	_encoding() (trans	<pre>detect_encoding() (translate.storage.xliff.xlifffile</pre>
	late.storage.properties.joomlafile method 578	, detect_header() (in module trans- late.storage.csvl10n), 425
	late.storage.properties.propfile method 580	- Dialect (class in translate.storage.ini), 446 , Dialect (class in translate.storage.properties), 570 DialectDefault (class in translate.storage.ini), 446
detect_	_encoding() (translate.storage.properties.stringsfile method 585	, o 1 1
detect_	_encoding() (translate.storage.properties.stringsutf8file method 587	
detect_	_encoding() (translate.storage.pypo.pofil method), 589	
detect	_encoding() (translate.storage.qm.qmfil method), 595	· · · · · · · · · · · · · · · · · · ·
detect_	_encoding() (translate.storage.qph.QphFilmethod), 600	•
detect_	_encoding() (translate.storage.rc.rcfil method), 606	· · · · · · · · · · · · · · · · · · ·
detect_	_encoding() (trans	

573		method), 435	
DialectStrings (class in	trans-		(trans-
late.storage.properties), 573		late.storage.html.POHTMLParser	method),
DialectStringsUtf8 (class in	trans-	432	
late.storage.properties), 574	0 61 1	Document (class in translate.misc.ourdom), 39	
dialogsizes() (translate.filters.checks.L20 method), 298	InChecker	DOMImplementation (class in late.misc.ourdom), 393	trans-
dialogsizes()	(trans-	doreplace() (translate.convert.convert	.Replacer
late.filters.checks.MozillaChecker 316	method),	method), 240 DotsProgressBar (class in	trans-
dict (translate.storage.catkeys.CatkeysUnit	attribute),	late.misc.progressbar), 394	ırans-
417		doublequoting()	(trans-
dict (translate.storage.omegat.OmegaTUnit 492	attribute),	late.filters.checks.CCLicenseChecker 270	method),
$\verb"dict" (translate.storage.utx. Utx Unit attribute),$		<pre>doublequoting()</pre>	(trans-
dict (translate.storage.wordfast.WordfastUtribute), 669	Init at-	late.filters.checks.DrupalChecker 276	method),
DirDiffer (class in translate.tools.pydiff), 6	97	doublequoting()	(trans-
Directory (class in translate.storage.director disable_interspersed_args()		late.filters.checks.GnomeChecker 281	method),
late.convert.convert.ArchiveConvertC	*		(trans-
method), 234	<i>F</i>	late.filters.checks.IOSChecker method	
disable_interspersed_args()	(trans-	doublequoting()	(trans-
late.convert.convert.ConvertOptionPa method), 237	ırser	late.filters.checks.KdeChecker 293	method),
disable_interspersed_args()	(trans-	doublequoting()	(trans-
late.convert.po2moz.MozConvertOpti method), 249	onParser	late.filters.checks.L20nChecker 299	method),
disable_interspersed_args()	(trans-	<pre>doublequoting()</pre>	(trans-
late.convert.po2tmx.TmxOptionParse method), 254	r	late.filters.checks.LibreOfficeChecker 304	method),
disable_interspersed_args()	(trans-	doublequoting()	(trans-
late.convert.po2wordfast.WfOptionPa method), 259	ırser	late.filters.checks.MinimalChecker 310	method),
disable_interspersed_args()	(trans-	<pre>doublequoting()</pre>	(trans-
late.filters.pofilter.FilterOptionParser 349	method),	late.filters.checks.MozillaChecker 316	method),
<pre>disable_interspersed_args()</pre>	(trans-	doublequoting()	(trans-
late.misc.optrecurse.RecursiveOption method), 390	Parser	late.filters.checks.OpenOfficeChecker 322	method),
disable_interspersed_args()	(trans-	doublequoting()	(trans-
late.tools.poconflicts.ConflictOptionF method), 683		late.filters.checks.ReducedChecker 327	method),
disable_interspersed_args()	(trans-	doublequoting()	(trans-
late.tools.pogrep.GrepOptionParser	method),	late.filters.checks.StandardChecker	method),
disable_interspersed_args()	(trans-	doublequoting()	(trans-
late.tools.porestructure.SplitOptionPomethod), 690		late.filters.checks.TermChecker	method),
disable_interspersed_args()	(trans-	<pre>doublespacing()</pre>	(trans-
late.tools.poterminology.Terminology method), 694			`
distance() (in module translate.search.lsht	ein). 405	doublespacing()	(trans-
do_encoding() (translate.storage.htm		late.filters.checks.DrupalChecker	method),

276		doublewords()	(trans-
doublespacing()  late.filters.checks.GnomeChecker	(trans- method),	late.filters.checks.MozillaChecker 316	method),
281 doublespacing() late.filters.checks.IOSChecker method	(trans-	doublewords()  late.filters.checks.OpenOfficeChecker 322	(trans- method),
doublespacing()  late.filters.checks.KdeChecker 293	(trans- method),	doublewords()  late.filters.checks.ReducedChecker  328	(trans- method),
doublespacing()  late.filters.checks.L20nChecker 299	(trans- method),	doublewords()  late.filters.checks.StandardChecker  333	(trans- method),
doublespacing()  late.filters.checks.LibreOfficeChecker 304	(trans- method),	doublewords () (translate.filters.checks.Tern method), 340 DrupalChecker (class in translate.filters.che	
doublespacing()  late.filters.checks.MinimalChecker 310	(trans- method),	dtdfile (class in translate.storage.dtd), 426 dtdunit (class in translate.storage.dtd), 428	, end), 270
doublespacing()  late.filters.checks.MozillaChecker  316	(trans- method),	E el (class in translate.lang.el), 364	
doublespacing()	(trans- method),	elem_at_offset()	(trans- method),
322 doublespacing() late.filters.checks.ReducedChecker 327	(trans- method),	elem_at_offset()	(trans- method),
doublespacing()  late.filters.checks.StandardChecker	(trans- method),	elem_at_offset()	(trans- method),
333 doublespacing()     late.filters.checks.TermChecker	(trans- method),	elem_at_offset()	(trans- method),
340 doublewords() late.filters.checks.CCLicenseChecker	(trans- method),	elem_at_offset()	(trans- method),
270 doublewords() late.filters.checks.DrupalChecker	(trans- method),	elem_at_offset()	(trans- method),
276 doublewords()     late.filters.checks.GnomeChecker	(trans- method),	elem_at_offset()	(trans- method),
282 doublewords() (translate.filters.checks.IC method), 287	OSChecker	elem_at_offset()  late.storage.placeables.base.Sub  510	(trans- method),
doublewords () (translate.filters.checks.Ka method), 293		<pre>elem_at_offset()     late.storage.placeables.base.X</pre>	(trans- method),
doublewords () (translate.filters.checks.L20 method), 299 doublewords ()	nChecker (trans-	509 elem_at_offset()	(trans-
late.filters.checks.LibreOfficeChecker 305		<pre>late.storage.placeables.general.AltAtt     method), 512 elem_at_offset()</pre>	rPiaceabie (trans-
doublewords()  late.filters.checks.MinimalChecker 310	(trans- method),	late.storage.placeables.general.XMLE method), 514	`

elem_at_offset()	(trans-	elem_	offset()(translate.storage.placed	ıbles.base.Bx
late.storage.placeables.general.XMD	LTagPlaceab	ole	method), 506	
method), 516		elem_	offset()	(trans-
elem_at_offset()	(trans-		late.storage.placeables.base.Ept	method),
$late.storage.placeables.interfaces.Box{\it a}$	asePlaceable		499	
method), 517		elem_	offset()(translate.storage.placed	ıbles.base.Ex
elem_at_offset()	(trans-		method), 507	
late.storage.placeables.interfaces.In method), 519	visiblePlace	<i>ab</i> dèem_	offset() (translate.storage.place method), 504	ables.base.G
elem_at_offset()	(trans-	elem	offset() (translate.storage.place	ables.base.It
late.storage.placeables.interfaces.M			method), 502	
method), 521	Ö		offset()	(trans-
elem_at_offset()	(trans-		late.storage.placeables.base.Ph	method),
late.storage.placeables.interfaces.Re	•	laceable		<i>"</i>
method), 522	•		offset()	(trans-
elem_at_offset()	(trans-	_	late.storage.placeables.base.Sub	method),
late.storage.placeables.interfaces.Si	•	able	510	//
method), 524	3		offset() (translate.storage.place	ables.base.X
elem_at_offset()	(trans-	_	method), 509	
late.storage.placeables.strelem.Strin	`	elem	offset()	(trans-
method), 526	3	_	late.storage.placeables.general.Ali	*
elem_at_offset()	(trans-		method), 512	
late.storage.placeables.terminology.	`	Płacembi		(trans-
method), 528	201111110108)	1 420tto_	late.storage.placeables.general.XN	*
elem_at_offset()	(trans-		method), 514	
late.storage.placeables.xliff.Bpt	method),	elem	offset()	(trans-
530	,,		late.storage.placeables.general.XN	•
elem_at_offset()	(trans-		method), 516	
late.storage.placeables.xliff.Bx	method),	elem	offset()	(trans-
535	,		late.storage.placeables.interfaces.i	*
elem_at_offset()	(trans-		method), 517	
late.storage.placeables.xliff.Ept	method),	elem	offset()	(trans-
532	,		late.storage.placeables.interfaces.i	,
elem_at_offset()	(trans-		method), 519	
late.storage.placeables.xliff.Ex	method),	elem	offset()	(trans-
537	,		late.storage.placeables.interfaces.i	`
elem_at_offset()	(trans-		method), 521	6
late.storage.placeables.xliff.G metho	•	elem	offset()	(trans-
elem_at_offset()	(trans-		late.storage.placeables.interfaces.i	,
late.storage.placeables.xliff.It metho	`		method), 522	tacement taceacte
elem_at_offset()	(trans-	elem	offset()	(trans-
late.storage.placeables.xliff.Ph	method),	010	late.storage.placeables.interfaces.	`
543	,,		method), 524	, , , , , , , , , , , , , , , , , , ,
elem_at_offset()	(trans-	elem	offset()	(trans-
late.storage.placeables.xliff.Sub	method),	_	late.storage.placeables.strelem.Str	,
542	,,,		method), 526	
elem_at_offset()	(trans-	elem	offset()	(trans-
late.storage.placeables.xliff.Unknow	`		late.storage.placeables.terminolog	`
method), 545			method), 529	)···
elem_at_offset()	(trans-	elem	offset()	(trans-
late.storage.placeables.xliff.X metho	`		late.storage.placeables.xliff.Bpt	method),
elem_offset()	(trans-		530	,
late.storage.placeables.base.Bpt	method),	elem	offset() (translate.storage.place	ables.xliff.Bx
497	,,	_	method), 535	<b></b>

method), 237 elem offset() (translate.storage.placeables.xliff.Ept method), enable\_interspersed\_args() (translate.convert.po2moz.MozConvertOptionParser elem\_offset() (translate.storage.placeables.xliff.Ex method), 249 *method*), 537 enable interspersed args() (transelem offset() (translate.storage.placeables.xliff.G late.convert.po2tmx.TmxOptionParser method), 539 method), 255 enable\_interspersed\_args() elem\_offset() (translate.storage.placeables.xliff.It (transmethod), 540 late.convert.po2wordfast.WfOptionParser elem\_offset() (translate.storage.placeables.xliff.Ph method), 259 *method*), 543 enable\_interspersed\_args() (transelem\_offset() late.filters.pofilter.FilterOptionParser method), (translate.storage.placeables.xliff.Sub method), 542 (transenable\_interspersed\_args() elem\_offset() (translate.misc.optrecurse.RecursiveOptionParser late.storage.placeables.xliff.UnknownXML *method*), 391 enable\_interspersed\_args() *method*), 545 (transelem\_offset() (translate.storage.placeables.xliff.X late.tools.poconflicts.ConflictOptionParser method), 534 method), 683 Element (class in translate.misc.ourdom), 393 enable interspersed args() (trans-ElementNotFoundError, 526 late.tools.pogrep.GrepOptionParser method), emails() (translate.filters.checks.CCLicenseChecker method), 270 enable\_interspersed\_args() (trans-(translate.filters.checks.DrupalChecker late.tools.porestructure.SplitOptionParser emails() *method*), 691 method), 276 emails() (translate.filters.checks.GnomeChecker enable\_interspersed\_args() method), 282 late. tools. poterminology. Terminology Option Parser(translate.filters.checks.IOSChecker emails() method), 694 method), 287 (translate.misc.multistring.multistring encode() emails() (translate.filters.checks.KdeChecker method), 388 method), 293 encode() (translate.storage.placeables.base.Bpt emails() (translate.filters.checks.L20nChecker method), 498 method), 299 encode() (translate.storage.placeables.base.Bx (translate.filters.checks.LibreOfficeChecker method), 506 emails() method), 305 (translate.storage.placeables.base.Ept encode() (translate.filters.checks.MinimalChecker method), 499 emails() encode() (translate.storage.placeables.base.Ex (translate.filters.checks.MozillaChecker method), 507 emails() method), 316 (translate.storage.placeables.base.G encode() emails() (translate.filters.checks.OpenOfficeChecker method), 504 (translate.storage.placeables.base.It method), 322 encode() (translate.filters.checks.ReducedChecker method), 502 emails() (translate.storage.placeables.base.Ph *method*), 328 encode() (translate. filters. checks. Standard Checkermethod), 501 emails() method), 333 encode() (translate.storage.placeables.base.Sub (translate.filters.checks.TermChecker emails() method), 511 method), 341 encode() (translate.storage.placeables.base.X emptyfiletotals() module (in transmethod), 509 late.storage.statsdb), 612 encode() (translate.storage.placeables.general.AltAttrPlaceable enable\_interspersed\_args() (transmethod), 512  $late. convert. Archive Convert Option Parser \verb|encode|| () (translate. storage. placeables. general. XMLEntity Placeable ()) (translate. storage. general. XMLEntity Placeable ()) (translate. storage. general. XMLEntity Placeable ()) (translate. general. xml. general. general. general. xml. general. general$ method), 234 method), 514 enable\_interspersed\_args() encode() (translate.storage.placeables.general.XMLTagPlaceable (translate.convert.convert.ConvertOptionParser method), 516

$\verb encode   (\textit{translate.storage.placeables.interfaces.BaseP }$	~
method), 518	endpunc() (translate.filters.checks.CCLicenseChecker
$\verb encode   (\textit{translate.storage.placeables.interfaces.Invisible}) $	
method), 519	endpunc() (translate.filters.checks.DrupalChecker
encode () (translate.storage.placeables.interfaces.Maskin	
method), 521	endpunc() (translate.filters.checks.GnomeChecker
encode () (translate.storage.placeables.interfaces.Replaceables.interfaces.Replaceables.action)	
method), 523	endpunc() (translate.filters.checks.IOSChecker
encode() (translate.storage.placeables.interfaces.Subflo	
method), 524	endpunc() (translate.filters.checks.KdeChecker
encode() (translate.storage.placeables.strelem.StringEle	
method), 526	endpunc() (translate.filters.checks.L20nChecker
encode() (translate.storage.placeables.terminology.Term	
method), 529	endpunc() (translate.filters.checks.LibreOfficeChecker
encode() (translate.storage.placeables.xliff.Bpt	method), 305
method), 531	endpunc() (translate.filters.checks.MinimalChecker
encode() (translate.storage.placeables.xliff.Bx	method), 311
method), 535	endpunc() (translate.filters.checks.MozillaChecker
encode() (translate.storage.placeables.xliff.Ept	method), 316
method), 532	endpunc () (translate.filters.checks.OpenOfficeChecker
encode() (translate.storage.placeables.xliff.Ex	method), 322
method), 537	endpunc() (translate.filters.checks.ReducedChecker
encode() (translate.storage.placeables.xliff.G	method), 328
method), 539	endpunc () (translate.filters.checks.StandardChecker method), 334
encode () (translate.storage.placeables.xliff.It method), 540	
	endpunc() (translate.filters.checks.TermChecker method), 341
encode() (translate.storage.placeables.xliff.Ph	memoa), 541
mathad) 511	
method), 544	endswith() (translate.misc.multistring.multistring
encode() (translate.storage.placeables.xliff.Sub	endswith() (translate.misc.multistring.multistring method), 388
encode() (translate.storage.placeables.xliff.Sub method), 542	endswith() (translate.misc.multistring.multistring method), 388 endwhitespace() (trans-
encode() (translate.storage.placeables.xliff.Sub method), 542 encode() (translate.storage.placeables.xliff.UnknownXM	endswith() (translate.misc.multistring.multistring method), 388 endwhitespace() (trans- AL late.filters.checks.CCLicenseChecker method),
encode () (translate.storage.placeables.xliff.Sub method), 542 encode () (translate.storage.placeables.xliff.UnknownXM method), 545	endswith() (translate.misc.multistring.multistring method), 388 endwhitespace() (trans- IL late.filters.checks.CCLicenseChecker method), 270
encode () (translate.storage.placeables.xliff.Sub method), 542 encode () (translate.storage.placeables.xliff.UnknownXM method), 545 encode () (translate.storage.placeables.xliff.X method),	endswith() (translate.misc.multistring.multistring method), 388 endwhitespace() (trans- AL late.filters.checks.CCLicenseChecker method), 270 endwhitespace() (trans-
encode () (translate.storage.placeables.xliff.Sub method), 542 encode () (translate.storage.placeables.xliff.UnknownXM method), 545 encode () (translate.storage.placeables.xliff.X method), 534	endswith() (translate.misc.multistring.multistring method), 388 endwhitespace() (trans- ML late.filters.checks.CCLicenseChecker method), 270 endwhitespace() (translate.filters.checks.DrupalChecker method),
encode () (translate.storage.placeables.xliff.Sub method), 542 encode () (translate.storage.placeables.xliff.UnknownXM method), 545 encode () (translate.storage.placeables.xliff.X method), 534 encode () (translate.storage.properties.Dialect class	endswith() (translate.misc.multistring.multistring method), 388 endwhitespace() (trans- ML late.filters.checks.CCLicenseChecker method), 270 endwhitespace() (trans- late.filters.checks.DrupalChecker method), 276
encode () (translate.storage.placeables.xliff.Sub method), 542 encode () (translate.storage.placeables.xliff.UnknownXM method), 545 encode () (translate.storage.placeables.xliff.X method), 534 encode () (translate.storage.properties.Dialect class method), 570	endswith() (translate.misc.multistring.multistring method), 388 endwhitespace() (trans- ML late.filters.checks.CCLicenseChecker method), 270 endwhitespace() (trans- late.filters.checks.DrupalChecker method), 276 endwhitespace() (trans-
encode () (translate.storage.placeables.xliff.Sub method), 542 encode () (translate.storage.placeables.xliff.UnknownXM method), 545 encode () (translate.storage.placeables.xliff.X method), 534 encode () (translate.storage.properties.Dialect class method), 570 encode () (translate.storage.properties.DialectFlex	endswith () (translate.misc.multistring.multistring method), 388 endwhitespace () (trans-  **ML late.filters.checks.CCLicenseChecker method), 270 endwhitespace () (trans- **late.filters.checks.DrupalChecker method), 276 endwhitespace () (trans- **late.filters.checks.GnomeChecker method), 276
encode () (translate.storage.placeables.xliff.Submethod), 542 encode () (translate.storage.placeables.xliff.UnknownXMmethod), 545 encode () (translate.storage.placeables.xliff.X method), 534 encode () (translate.storage.properties.Dialect classmethod), 570 encode () (translate.storage.properties.DialectFlexclassmethod), 570	endswith() (translate.misc.multistring.multistring method), 388 endwhitespace() (trans- AL late.filters.checks.CCLicenseChecker method), 270 endwhitespace() (trans- late.filters.checks.DrupalChecker method), 276 endwhitespace() (trans- late.filters.checks.GnomeChecker method), 282
encode () (translate.storage.placeables.xliff.Sub method), 542 encode () (translate.storage.placeables.xliff.UnknownXM method), 545 encode () (translate.storage.placeables.xliff.X method), 534 encode () (translate.storage.properties.Dialect class method), 570 encode () (translate.storage.properties.DialectFlex class method), 570 encode () (translate.storage.properties.DialectGaia	endswith() (translate.misc.multistring.multistring method), 388 endwhitespace() (trans- AL late.filters.checks.CCLicenseChecker method), 270 endwhitespace() (trans- late.filters.checks.DrupalChecker method), 276 endwhitespace() (trans- late.filters.checks.GnomeChecker method), 282 endwhitespace() (trans-
encode () (translate.storage.placeables.xliff.Submethod), 542 encode () (translate.storage.placeables.xliff.UnknownXMmethod), 545 encode () (translate.storage.placeables.xliff.X method), 534 encode () (translate.storage.properties.Dialect classmethod), 570 encode () (translate.storage.properties.DialectFlexclassmethod), 570 encode () (translate.storage.properties.DialectGaiaclassmethod), 571	endswith() (translate.misc.multistring.multistring method), 388 endwhitespace() (trans-ML late.filters.checks.CCLicenseChecker method), 270 endwhitespace() (translate.filters.checks.DrupalChecker method), 276 endwhitespace() (translate.filters.checks.GnomeChecker method), 282 endwhitespace() (translate.filters.checks.GnomeChecker method), 282 endwhitespace() (translate.filters.checks.IOSChecker method), 288
encode () (translate.storage.placeables.xliff.Sub method), 542 encode () (translate.storage.placeables.xliff.UnknownXM method), 545 encode () (translate.storage.placeables.xliff.X method), 534 encode () (translate.storage.properties.Dialect class method), 570 encode () (translate.storage.properties.DialectFlex class method), 570 encode () (translate.storage.properties.DialectGaia class method), 571 encode () (translate.storage.properties.DialectJava	endswith() (translate.misc.multistring.multistring method), 388 endwhitespace() (trans- AL late.filters.checks.CCLicenseChecker method), 270 endwhitespace() (trans- late.filters.checks.DrupalChecker method), 276 endwhitespace() (trans- late.filters.checks.GnomeChecker method), 282 endwhitespace() (trans- late.filters.checks.IOSChecker method), 288 endwhitespace() (trans-
encode () (translate.storage.placeables.xliff.Submethod), 542 encode () (translate.storage.placeables.xliff.UnknownXMmethod), 545 encode () (translate.storage.placeables.xliff.X method), 534 encode () (translate.storage.properties.Dialect classmethod), 570 encode () (translate.storage.properties.DialectFlexclassmethod), 570 encode () (translate.storage.properties.DialectGaiaclassmethod), 571 encode () (translate.storage.properties.DialectJavaclassmethod), 571	endswith() (translate.misc.multistring.multistring method), 388 endwhitespace() (trans-ML late.filters.checks.CCLicenseChecker method), 270 endwhitespace() (translate.filters.checks.DrupalChecker method), 276 endwhitespace() (translate.filters.checks.GnomeChecker method), 282 endwhitespace() (translate.filters.checks.GnomeChecker method), 282 endwhitespace() (translate.filters.checks.IOSChecker method), 288
encode () (translate.storage.placeables.xliff.Sub method), 542 encode () (translate.storage.placeables.xliff.UnknownXM method), 545 encode () (translate.storage.placeables.xliff.X method), 534 encode () (translate.storage.properties.Dialect class method), 570 encode () (translate.storage.properties.DialectFlex class method), 570 encode () (translate.storage.properties.DialectGaia class method), 571 encode () (translate.storage.properties.DialectJava class method), 571 encode () (translate.storage.properties.DialectJavaUtf8	endswith() (translate.misc.multistring.multistring method), 388 endwhitespace() (trans- AL late.filters.checks.CCLicenseChecker method), 270 endwhitespace() (trans- late.filters.checks.DrupalChecker method), 276 endwhitespace() (trans- late.filters.checks.GnomeChecker method), 282 endwhitespace() (trans- late.filters.checks.IOSChecker method), 288 endwhitespace() (trans- late.filters.checks.KdeChecker method), 288 endwhitespace() (trans- late.filters.checks.KdeChecker method), 294
encode () (translate.storage.placeables.xliff.Submethod), 542 encode () (translate.storage.placeables.xliff.UnknownXMmethod), 545 encode () (translate.storage.placeables.xliff.X method), 534 encode () (translate.storage.properties.Dialect classmethod), 570 encode () (translate.storage.properties.DialectFlexclassmethod), 570 encode () (translate.storage.properties.DialectGaiaclassmethod), 571 encode () (translate.storage.properties.DialectJavaclassmethod), 571 encode () (translate.storage.properties.DialectJavaclassmethod), 571 encode () (translate.storage.properties.DialectJavaUtf8classmethod), 572	endswith() (translate.misc.multistring.multistring method), 388 endwhitespace() (trans- AL late.filters.checks.CCLicenseChecker method), 270 endwhitespace() (trans- late.filters.checks.DrupalChecker method), 276 endwhitespace() (trans- late.filters.checks.GnomeChecker method), 282 endwhitespace() (trans- late.filters.checks.IOSChecker method), 288 endwhitespace() (trans- late.filters.checks.KdeChecker method), 288 endwhitespace() (trans- late.filters.checks.KdeChecker method), 294 endwhitespace() (trans-
encode () (translate.storage.placeables.xliff.Sub method), 542 encode () (translate.storage.placeables.xliff.UnknownXM method), 545 encode () (translate.storage.placeables.xliff.X method), 534 encode () (translate.storage.properties.Dialect class method), 570 encode () (translate.storage.properties.DialectFlex class method), 570 encode () (translate.storage.properties.DialectGaia class method), 571 encode () (translate.storage.properties.DialectJava class method), 571 encode () (translate.storage.properties.DialectJavaUtf8 class method), 572 encode () (translate.storage.properties.DialectJoomla	endswith() (translate.misc.multistring.multistring method), 388 endwhitespace() (trans- AL late.filters.checks.CCLicenseChecker method), 270 endwhitespace() (trans- late.filters.checks.DrupalChecker method), 276 endwhitespace() (trans- late.filters.checks.GnomeChecker method), 282 endwhitespace() (trans- late.filters.checks.IOSChecker method), 288 endwhitespace() (trans- late.filters.checks.KdeChecker method), 288 endwhitespace() (trans- late.filters.checks.KdeChecker method), 294
encode () (translate.storage.placeables.xliff.Sub method), 542 encode () (translate.storage.placeables.xliff.UnknownXM method), 545 encode () (translate.storage.placeables.xliff.X method), 534 encode () (translate.storage.properties.Dialect class method), 570 encode () (translate.storage.properties.DialectFlex class method), 570 encode () (translate.storage.properties.DialectGaia class method), 571 encode () (translate.storage.properties.DialectJava class method), 571 encode () (translate.storage.properties.DialectJavaUtf8 class method), 572 encode () (translate.storage.properties.DialectJoomla class method), 572	endswith() (translate.misc.multistring.multistring method), 388 endwhitespace() (trans- AL late.filters.checks.CCLicenseChecker method), 270 endwhitespace() (trans- late.filters.checks.DrupalChecker method), 276 endwhitespace() (trans- late.filters.checks.GnomeChecker method), 282 endwhitespace() (trans- late.filters.checks.IOSChecker method), 288 endwhitespace() (trans- late.filters.checks.KdeChecker method), 294 endwhitespace() (trans- late.filters.checks.L2OnChecker method), 299
encode () (translate.storage.placeables.xliff.Sub method), 542 encode () (translate.storage.placeables.xliff.UnknownXM method), 545 encode () (translate.storage.placeables.xliff.X method), 534 encode () (translate.storage.properties.Dialect class method), 570 encode () (translate.storage.properties.DialectFlex class method), 570 encode () (translate.storage.properties.DialectGaia class method), 571 encode () (translate.storage.properties.DialectJava class method), 571 encode () (translate.storage.properties.DialectJavaUtf8 class method), 572 encode () (translate.storage.properties.DialectJoomla class method), 572 encode () (translate.storage.properties.DialectJoomla class method), 572 encode () (translate.storage.properties.DialectMozilla	endswith () (translate.misc.multistring.multistring method), 388 endwhitespace () (trans- ML late.filters.checks.CCLicenseChecker method), 270 endwhitespace () (trans- late.filters.checks.DrupalChecker method), 276 endwhitespace () (trans- late.filters.checks.GnomeChecker method), 282 endwhitespace () (trans- late.filters.checks.IOSChecker method), 288 endwhitespace () (trans- late.filters.checks.KdeChecker method), 294 endwhitespace () (trans- late.filters.checks.L2OnChecker method), 299 endwhitespace () (trans- late.filters.checks.L2OnChecker method), 299 endwhitespace () (trans-
encode () (translate.storage.placeables.xliff.Sub method), 542 encode () (translate.storage.placeables.xliff.UnknownXM method), 545 encode () (translate.storage.placeables.xliff.X method), 534 encode () (translate.storage.properties.Dialect class method), 570 encode () (translate.storage.properties.DialectFlex class method), 570 encode () (translate.storage.properties.DialectGaia class method), 571 encode () (translate.storage.properties.DialectJava class method), 571 encode () (translate.storage.properties.DialectJavaUtf8 class method), 572 encode () (translate.storage.properties.DialectJoomla class method), 572 encode () (translate.storage.properties.DialectMozilla class method), 572	endswith() (translate.misc.multistring.multistring method), 388 endwhitespace() (trans- AL late.filters.checks.CCLicenseChecker method), 270 endwhitespace() (trans- late.filters.checks.DrupalChecker method), 276 endwhitespace() (trans- late.filters.checks.GnomeChecker method), 282 endwhitespace() (trans- late.filters.checks.IOSChecker method), 288 endwhitespace() (trans- late.filters.checks.KdeChecker method), 294 endwhitespace() (trans- late.filters.checks.L2OnChecker method), 299
encode () (translate.storage.placeables.xliff.Sub method), 542 encode () (translate.storage.placeables.xliff.UnknownXM method), 545 encode () (translate.storage.placeables.xliff.X method), 534 encode () (translate.storage.properties.Dialect class method), 570 encode () (translate.storage.properties.DialectFlex class method), 570 encode () (translate.storage.properties.DialectGaia class method), 571 encode () (translate.storage.properties.DialectJava class method), 571 encode () (translate.storage.properties.DialectJavaUtf8 class method), 572 encode () (translate.storage.properties.DialectJoomla class method), 572 encode () (translate.storage.properties.DialectMozilla class method), 572 encode () (translate.storage.properties.DialectMozilla class method), 572	endswith () (translate.misc.multistring.multistring method), 388 endwhitespace () (translate.filters.checks.CCLicenseChecker method), 270 endwhitespace () (translate.filters.checks.DrupalChecker method), 276 endwhitespace () (translate.filters.checks.GnomeChecker method), 282 endwhitespace () (translate.filters.checks.IOSChecker method), 288 endwhitespace () (translate.filters.checks.KdeChecker method), 288 endwhitespace () (translate.filters.checks.KdeChecker method), 294 endwhitespace () (translate.filters.checks.L2OnChecker method), 299 endwhitespace () (translate.filters.checks.L2OnChecker method), 305
encode () (translate.storage.placeables.xliff.Sub method), 542 encode () (translate.storage.placeables.xliff.UnknownXM method), 545 encode () (translate.storage.placeables.xliff.X method), 534 encode () (translate.storage.properties.Dialect class method), 570 encode () (translate.storage.properties.DialectFlex class method), 570 encode () (translate.storage.properties.DialectGaia class method), 571 encode () (translate.storage.properties.DialectJava class method), 571 encode () (translate.storage.properties.DialectJavaUtf8 class method), 572 encode () (translate.storage.properties.DialectJoomla class method), 572 encode () (translate.storage.properties.DialectMozilla class method), 572 encode () (translate.storage.properties.DialectMozilla class method), 572 encode () (translate.storage.properties.DialectSkype class method), 573	endswith () (translate.misc.multistring.multistring method), 388 endwhitespace () (translate.filters.checks.CCLicenseChecker method), 270 endwhitespace () (translate.filters.checks.DrupalChecker method), 276 endwhitespace () (translate.filters.checks.GnomeChecker method), 282 endwhitespace () (translate.filters.checks.IOSChecker method), 288 endwhitespace () (translate.filters.checks.KdeChecker method), 288 endwhitespace () (translate.filters.checks.KdeChecker method), 294 endwhitespace () (translate.filters.checks.L2OnChecker method), 299 endwhitespace () (translate.filters.checks.LibreOfficeChecker method), 305 endwhitespace () (translate.filters.checks.LibreOfficeChecker method), 305
encode () (translate.storage.placeables.xliff.Sub method), 542 encode () (translate.storage.placeables.xliff.UnknownXM method), 545 encode () (translate.storage.placeables.xliff.X method), 534 encode () (translate.storage.properties.Dialect class method), 570 encode () (translate.storage.properties.DialectFlex class method), 570 encode () (translate.storage.properties.DialectGaia class method), 571 encode () (translate.storage.properties.DialectJava class method), 571 encode () (translate.storage.properties.DialectJavaUtf8 class method), 572 encode () (translate.storage.properties.DialectJoomla class method), 572 encode () (translate.storage.properties.DialectMozilla class method), 572 encode () (translate.storage.properties.DialectMozilla class method), 572 encode () (translate.storage.properties.DialectSkype class method), 573	endswith () (translate.misc.multistring.multistring method), 388 endwhitespace () (translate.filters.checks.CCLicenseChecker method), 270 endwhitespace () (translate.filters.checks.DrupalChecker method), 276 endwhitespace () (translate.filters.checks.GnomeChecker method), 282 endwhitespace () (translate.filters.checks.IOSChecker method), 288 endwhitespace () (translate.filters.checks.KdeChecker method), 288 endwhitespace () (translate.filters.checks.KdeChecker method), 294 endwhitespace () (translate.filters.checks.L2OnChecker method), 299 endwhitespace () (translate.filters.checks.LibreOfficeChecker method), 305 endwhitespace () (translate.filters.checks.LibreOfficeChecker method), 305
encode () (translate.storage.placeables.xliff.Sub method), 542 encode () (translate.storage.placeables.xliff.UnknownXM method), 545 encode () (translate.storage.placeables.xliff.X method), 534 encode () (translate.storage.properties.Dialect class method), 570 encode () (translate.storage.properties.DialectFlex class method), 570 encode () (translate.storage.properties.DialectGaia class method), 571 encode () (translate.storage.properties.DialectJava class method), 571 encode () (translate.storage.properties.DialectJavaUtf8 class method), 572 encode () (translate.storage.properties.DialectJoomla class method), 572 encode () (translate.storage.properties.DialectMozilla class method), 572 encode () (translate.storage.properties.DialectSkype class method), 573 encode () (translate.storage.properties.DialectStrings	endswith () (translate.misc.multistring.multistring method), 388 endwhitespace () (translate.filters.checks.CCLicenseChecker method), 270 endwhitespace () (translate.filters.checks.DrupalChecker method), 276 endwhitespace () (translate.filters.checks.GnomeChecker method), 282 endwhitespace () (translate.filters.checks.IOSChecker method), 288 endwhitespace () (translate.filters.checks.KdeChecker method), 288 endwhitespace () (translate.filters.checks.KdeChecker method), 294 endwhitespace () (translate.filters.checks.L2OnChecker method), 299 endwhitespace () (translate.filters.checks.LibreOfficeChecker method), 305 endwhitespace () (translate.filters.checks.LibreOfficeChecker method), 305 endwhitespace () (translate.filters.checks.MinimalChecker method), 311

317	$\verb escapes()  \textit{(translate.filters.checks.CCL icenseChecker} $
endwhitespace() (trans-	method), 270
late.filters.checks.OpenOfficeChecker method), 322	escapes() (translate.filters.checks.DrupalChecker method), 276
endwhitespace() (trans-	escapes() (translate.filters.checks.GnomeChecker
late.filters.checks.ReducedChecker method),	method), 282
328	escapes() (translate.filters.checks.IOSChecker
endwhitespace() (trans-	method), 288
late.filters.checks.StandardChecker method), 334	escapes() (translate.filters.checks.KdeChecker method), 294
endwhitespace() (trans-	escapes() (translate.filters.checks.L20nChecker
late.filters.checks.TermChecker method),	method), 300
341	escapes() (translate.filters.checks.LibreOfficeChecker
<pre>entitydecode() (in module translate.misc.quote),</pre>	method), 305
394	escapes() (translate.filters.checks.MinimalChecker
<pre>entityencode() (in module translate.misc.quote),</pre>	method), 311
394	escapes() (translate.filters.checks.MozillaChecker
Ept (class in translate.storage.placeables.base), 499	method), 317
Ept (class in translate.storage.placeables.xliff), 532	escapes() (translate.filters.checks.OpenOfficeChecker
error() (translate.convert.ArchiveConvertOption	
method), 234	escapes() (translate.filters.checks.ReducedChecker
error() (translate.convert.convert.ConvertOptionParser	method), 328
method), 237	escapes() (translate.filters.checks.StandardChecker
error() (translate.convert.po2moz.MozConvertOptionPa	- · · · · · · · · · · · · · · · · · · ·
method), 249	escapes() (translate.filters.checks.TermChecker
error() (translate.convert.po2tmx.TmxOptionParser	method), 341
method), 255	ethiopicpunc (translate.lang.common.Common at-
<pre>error() (translate.convert.po2wordfast.WfOptionParser</pre>	tribute), 358
method), 259	Ex (class in translate.storage.placeables.base), 507
error() (translate.filters.pofilter.FilterOptionParser	Ex (class in translate.storage.placeables.xliff), 536
method), 349	<pre>expandtabs() (translate.misc.multistring.multistring</pre>
${\tt error}$ () (translate.misc.optrecurse.RecursiveOptionParse	er method), 388
method), 391	expansion_factors (in module translate.lang.data),
$\verb error   () (translate.tools.poconflicts.ConflictOptionParset$	360
method), 683	ExpatBuilderNS (class in translate.misc.ourdom),
error() (translate.tools.pogrep.GrepOptionParser	393
method), 687	<pre>export_file() (translate.storage.project.Project</pre>
$\verb error  ()   \textit{(translate.tools.porestructure.SplitOptionParser}$	
method), 691	<pre>extendtm() (translate.search.match.matcher method),</pre>
$\verb error   (translate.tools.poterminology.TerminologyOptical and all the state of $	
method), 694	$\verb  extendtm()  \textit{(translate.search.match.terminologymatcher)  }$
es (class in translate.lang.es), 364	method), 406
escape() (in module translate.storage.trados), 640	
escape_help_text() (in module trans-	Extensions (translate.storage.base.TranslationStore
	${\tt Extensions} \  \  (\textit{translate.storage.base.TranslationStore} \\ \textit{attribute}), 408$
late.storage.oo), 494	$ \begin{array}{c} {\tt Extensions} & \textit{(translate.storage.base.TranslationStore} \\ & \textit{attribute)}, 408 \\ {\tt extract()} & \textit{(in module translate.convert.accesskey)}, \end{array} $
late.storage.oo), 494 escape_text() (in module translate.storage.oo), 494	Extensions (translate.storage.base.TranslationStore attribute), 408 extract() (in module translate.convert.accesskey), 233
late.storage.oo), 494 escape_text() (in module translate.storage.oo), 494 escape_to_python() (in module trans-	Extensions (translate.storage.base.TranslationStore attribute), 408 extract() (in module translate.convert.accesskey), 233 extract() (in module translate.misc.quote), 395
late.storage.oo), 494 escape_text() (in module translate.storage.oo), 494 escape_to_python() (in module translate.storage.rc), 606	Extensions (translate.storage.base.TranslationStore attribute), 408  extract() (in module translate.convert.accesskey), 233  extract() (in module translate.misc.quote), 395  extract_msgid_comment() (in module trans-
late.storage.oo), 494 escape_text() (in module translate.storage.oo), 494 escape_to_python() (in module translate.storage.rc), 606 escape_to_rc() (in module translate.storage.rc),	Extensions (translate.storage.base.TranslationStore attribute), 408 extract() (in module translate.convert.accesskey), 233 extract() (in module translate.misc.quote), 395 extract_msgid_comment() (in module translate.storage.pocommon), 552
late.storage.oo), 494 escape_text() (in module translate.storage.oo), 494 escape_to_python() (in module translate.storage.rc), 606 escape_to_rc() (in module translate.storage.rc), 606	Extensions (translate.storage.base.TranslationStore attribute), 408  extract() (in module translate.convert.accesskey), 233  extract() (in module translate.misc.quote), 395  extract_msgid_comment() (in module translate.storage.pocommon), 552  extractpoline() (in module translate.storage.pocommon)
late.storage.oo), 494 escape_text() (in module translate.storage.oo), 494 escape_to_python() (in module translate.storage.rc), 606 escape_to_rc() (in module translate.storage.rc), 606 escapecontrols() (in module translate.misc.quote),	Extensions (translate.storage.base.TranslationStore attribute), 408  extract() (in module translate.convert.accesskey), 233  extract() (in module translate.misc.quote), 395  extract_msgid_comment() (in module translate.storage.pocommon), 552  extractpoline() (in module translate.storage.pypo), 588
late.storage.oo), 494 escape_text() (in module translate.storage.oo), 494 escape_to_python() (in module translate.storage.rc), 606 escape_to_rc() (in module translate.storage.rc), 606	Extensions (translate.storage.base.TranslationStore attribute), 408  extract() (in module translate.convert.accesskey), 233  extract() (in module translate.misc.quote), 395  extract_msgid_comment() (in module translate.storage.pocommon), 552  extractpoline() (in module translate.storage.pocommon)

F		fallback_detection()	(trans-
fa (class in translate.lang.fa), 365		late.storage.properties.javautf8file	method),
fallback_detection()	(trans-	577	
	nethod),	<pre>fallback_detection()</pre>	(trans-
408		late.storage.properties.joomlafile	method),
fallback_detection()	(trans-	578	
	nethod),	<pre>fallback_detection()</pre>	(trans-
415		late.storage.properties.propfile	method),
fallback_detection()	(trans-	580	
late.storage.csvl10n.csvfile method), 420		<pre>fallback_detection()</pre>	(trans-
fallback_detection()	(trans-	late.storage.properties.stringsfile	method),
late.storage.dtd.dtdfile method), 426	(irans	585	
fallback_detection()	(trans-	fallback_detection()	(trans-
late.storage.html.htmlfile method), 435	(irans	late.storage.properties.stringsutf8file	method),
fallback_detection()	(trans-	587	
	nethod),	fallback_detection()	(trans-
432	ieinou),	late.storage.pypo.pofile method), 589	
	(tuana	<pre>fallback_detection()</pre>	(trans-
fallback_detection()	(trans-	late.storage.qm.qmfile method), 595	(11 011 15
late.storage.ical.icalfile method), 441		fallback_detection()	(trans-
fallback_detection()	(trans-	late.storage.qph.QphFile method), 60	`
late.storage.ini.inifile method), 446		fallback_detection() (translate.storag	
fallback_detection()	(trans-	method), 606	e.rc.rcjiie
- ·	ıethod),		(trans-
452		fallback_detection()	`
fallback_detection()	(trans-	late.storage.subtitles.AdvSubStationA	ірпатие
$late.storage.jsonl 10 n. Json File\ method),$		method), 613	(tuans
fallback_detection()	(trans-	fallback_detection()	(trans-
late.storage.jsonl10n.JsonNestedFile m	ıethod),	late.storage.subtitles.MicroDVDFile 614	тетоа),
458			(tuana
fallback_detection()	(trans-	fallback_detection()	(trans-
late.storage.jsonl 10 n. Web Extension Json and the state of the sta	iFile	late.storage.subtitles.SubRipFile	method),
method), 466		616	
fallback_detection()	(trans-	fallback_detection()	(trans-
late.storage.lisa.LISAfile method), 471		late.storage.subtitles.SubStationAlpha	ıFile
fallback_detection()	(trans-	method), 618	,
late.storage.mo.mofile method), 477		fallback_detection()	(trans-
fallback_detection()	(trans-	late.storage.subtitles.SubtitleFile	method),
late.storage.mozilla_lang.LangStore m	nethod),	619	
483		<pre>fallback_detection()</pre>	(trans-
fallback_detection()	(trans-	late.storage.tbx.tbxfile method), 624	
	nethod),	<pre>fallback_detection()</pre>	(trans-
488	,,	late.storage.tiki.TikiStore method), 63	0
fallback_detection()	(trans-	fallback_detection()	(trans-
	nethod),	late.storage.tmx.tmxfile method), 635	
489	,,	fallback_detection()	(trans-
fallback_detection()	(trans-	late.storage.trados.TradosTxtTmFile	method),
late.storage.php.phpfile method), 547	(irans	644	
fallback_detection()	(trans-	<pre>fallback_detection() (translate.storage</pre>	e.ts2.tsfile
late.storage.pocommon.pofile method),	`	method), 645	3
		fallback_detection()	(trans-
fallback_detection()	(trans-	late.storage.txt.TxtFile method), 651	(
late.storage.poxliff.PoXliffFile method),		fallback_detection()	(trans-
fallback_detection()	(trans-	late.storage.utx.UtxFile method), 656	(ii aiii)
~ · · · ·	nethod),	fallback_detection()	(trans-
575		Tallback_acception()	(ii aiis-

$late. storage. word fast. Word fast TMF ile\ method),$	334
667	filepaths() (translate.filters.checks.TermChecker
<pre>fallback_detection() (trans-</pre>	method), 341
late.storage.xliff.xlifffile method), 673	filestatestats() (trans-
feed() (translate.storage.html.htmlfile method), 435	late.storage.statsdb.StatsCache method),
feed() (translate.storage.html.POHTMLParser	612
method), 432	filestats() (translate.storage.statsdb.StatsCache
fi (class in translate.lang.fi), 366	method), 612
field_analyzers (trans-	filetotals() (translate.storage.statsdb.StatsCache
late. search. in dexing. Common Indexer. Common Design C	
attribute), 398	filteraccelerators() (in module trans-
FIELDNAMES (in module translate.storage.catkeys), 419	late.filters.prefilters), 351
FIELDNAMES_HEADER (in module trans-	filteraccelerators_by_list() (trans-
late.storage.catkeys), 420	late.filters.checks.CCLicenseChecker method),
FIELDNAMES_HEADER_DEFAULTS (in module trans-	271
late.storage.catkeys), 420	filteraccelerators_by_list() (trans-
<pre>file_iter() (translate.storage.directory.Directory</pre>	late.filters.checks.DrupalChecker method), 277
<pre>file_iter() (translate.storage.zip.ZIPFile method),</pre>	filteraccelerators_by_list() (trans-
681	late.filters.checks.GnomeChecker method),
filechecks() (translate.storage.statsdb.StatsCache	282
method), 611	filteraccelerators_by_list() (trans-
FileDiffer (class in translate.tools.pydiff), 698	late.filters.checks.IOSChecker method), 288
FileExistsInProjectError, 568	filteraccelerators_by_list() (trans-
FileNotInProjectError, 568	late.filters.checks.KdeChecker method),
filepaths() (trans-	294
late.filters.checks.CCLicenseChecker method),	filteraccelerators_by_list() (trans-
271	late.filters.checks.L20nChecker method),
filepaths() (translate.filters.checks.DrupalChecker	300
method), 277	filteraccelerators_by_list() (trans-
filepaths() (translate.filters.checks.GnomeChecker method), 282	late.filters.checks.LibreOfficeChecker method), 305
filepaths() (translate.filters.checks.IOSChecker	filteraccelerators_by_list() (trans-
method), 288	late.filters.checks.MinimalChecker method),
filepaths() (translate.filters.checks.KdeChecker	311
method), 294	filteraccelerators_by_list() (trans-
filepaths() (translate.filters.checks.L2OnChecker method), 300	late.filters.checks.MozillaChecker method), 317
filepaths() (trans-	filteraccelerators_by_list() (trans-
late.filters.checks.LibreOfficeChecker method), 305	late.filters.checks.OpenOfficeChecker method), 323
filepaths() (trans-	filteraccelerators_by_list() (trans-
late.filters.checks.MinimalChecker method), 311	late.filters.checks.ReducedChecker method), 328
<pre>filepaths() (translate.filters.checks.MozillaChecker</pre>	<pre>filteraccelerators_by_list() (trans-</pre>
method), 317	late.filters.checks.StandardChecker method),
filepaths() (trans-	334
late.filters.checks.OpenOfficeChecker method),	filteraccelerators_by_list() (trans-
323	late. filters. checks. Standard Unit Checker
filepaths() (trans-	method), 338
late.filters.checks.ReducedChecker method),	filteraccelerators_by_list() (trans-
328	late.filters.checks.TermChecker method),
filepaths() (trans-	341
late.filters.checks.StandardChecker method),	filteraccelerators_by_list() (trans-

late.filters.checks.TranslationChecker method),	method), 250
345	finalizetempoutputfile() (trans-
filteraccelerators_by_list() (trans-late.filters.checks.UnitChecker method),	late.convert.po2tmx.TmxOptionParser method), 255
346	finalizetempoutputfile() (trans-
filtercount() (in module translate.filters.helpers), 348	late.convert.po2wordfast.WfOptionParser method), 259
FilterFailure, 280	finalizetempoutputfile() (trans-
<pre>filterinputformats() (trans- late.convert.convert.ArchiveConvertOptionParser</pre>	late.filters.pofilter.FilterOptionParser method), 349
method), 234	finalizetempoutputfile() (trans-
<pre>filterinputformats() (trans- late.convert.convert.ConvertOptionParser</pre>	late.misc.optrecurse.RecursiveOptionParser method), 391
method), 238	finalizetempoutputfile() (trans-
filterinputformats() (trans- late.convert.po2moz.MozConvertOptionParser	late.tools.poconflicts.ConflictOptionParser method), 683
method), 249 filterinputformats() (trans-	finalizetempoutputfile() (trans- late.tools.pogrep.GrepOptionParser method), 687
late.convert.po2tmx.TmxOptionParser	
<pre>method), 255 filterinputformats() (trans- late.convert.po2wordfast.WfOptionParser</pre>	finalizetempoutputfile() (trans- late.tools.porestructure.SplitOptionParser method), 691
method), 259	finalizetempoutputfile() (trans-
FilterOptionParser (class in trans- late.filters.pofilter), 348	late.tools.poterminology.TerminologyOptionParser method), 694
	find() (translate.misc.multistring.multistring method),
late.convert.convert.ArchiveConvertOptionParser	
<pre>method), 234 filteroutputoptions() (trans-</pre>	find() (translate.storage.placeables.base.Bpt method), 498
	find() (translate.storage.placeables.base.Bx method), 506
	find() (translate.storage.placeables.base.Ept method),
late.convert.po2moz.MozConvertOptionParser method), 249	499 find() (translate.storage.placeables.base.Ex method),
filteroutputoptions() (trans-	507
	find() (translate.storage.placeables.base.G method), 504
filteroutputoptions() (trans- late.convert.po2wordfast.WfOptionParser	find() (translate.storage.placeables.base.It method), 503
<pre>method), 259 filtertestmethod() (in module trans-</pre>	find() (translate.storage.placeables.base.Ph method), 501
late.filters.helpers), 348 filtervariables() (in module trans-	find() (translate.storage.placeables.base.Sub method), 511
<pre>late.filters.prefilters), 351 filterwordswithpunctuation() (in module</pre>	find() (translate.storage.placeables.base.X method), 509
<pre>translate.filters.prefilters), 352 finalizetempoutputfile() (trans-</pre>	find() (translate.storage.placeables.general.AltAttrPlaceable method), 512
	find() (translate.storage.placeables.general.XMLEntityPlaceabl method), 514
finalizetempoutputfile() (trans-	find() (translate.storage.placeables.general.XMLTagPlaceable
late.convert.convert.ConvertOptionParser method), 238	method), 516 find() (translate.storage.placeables.interfaces.BasePlaceable
finalizetempoutputfile() (trans-	method), 518
late.convert.po2moz.MozConvertOptionParser	find()(translate.storage.placeables.interfaces.InvisiblePlaceabl

	method), 519		late.storage.properties.DialectSkype	class
find()	(translate.storage.placeables.interfaces.)	MaskingF	Placeable method), 573	
	method), 521		<pre>find_delimiter()</pre>	(trans-
find()	(translate.storage.placeables.interfaces.)	Replacem	entPlacealbute.storage.properties.DialectStrings	class
	method), 523		method), 573	
find()	(translate.storage.placeables.interfaces.S	SubflowP	<i>lāceabl<u>e</u></i> delimiter()	(trans-
	method), 524		late.storage.properties.DialectStrings	Utf8
find()	(translate.storage.placeables.strelem.Str	ingElem	class method), 574	
	method), 527		<pre>find_dom_root() (in module</pre>	trans-
find()		y. <i>Termino</i>	ologyPlace <b>hhlæ</b> storage.xml_extract.generate), 67	9
	method), 529		<pre>find_elems_with()</pre>	(trans-
find()	(translate.storage.placeables.xliff.Bpt m	ethod),	late.storage.placeables.base.Bpt	method),
	531		498	
find()	(translate.storage.placeables.xliff.Bx m	ethod),	find_elems_with()	(trans-
	536		late.storage.placeables.base.Bx	method),
find()	(translate.storage.placeables.xliff.Ept m	ethod),	506	
	532		find_elems_with()	(trans-
find()	(translate.storage.placeables.xliff.Ex m 537	ethod),	late.storage.placeables.base.Ept 499	method),
find()	(translate.storage.placeables.xliff.G m	ethod),	find_elems_with()	(trans-
	539		late.storage.placeables.base.Ex	method),
find()	(translate.storage.placeables.xliff.It m	ethod),	507	
	540		find_elems_with()	(trans-
find()	(translate.storage.placeables.xliff.Ph m	ethod),	late.storage.placeables.base.G 504	method),
find()	(translate.storage.placeables.xliff.Sub m	ethod).	find_elems_with()	(trans-
- ()	542	,,,	late.storage.placeables.base.It	method),
find()	(translate.storage.placeables.xliff.Unkno	wnXML	503	,,,
.,	method), 545		find_elems_with()	(trans-
find()		ethod),	late.storage.placeables.base.Ph	method),
	534	,,	501	,,,
find_a	all () (in module translate.misc.quote), 3	395	find_elems_with()	(trans-
	delimiter()	(trans-	late.storage.placeables.base.Sub	method),
	late.storage.properties.Dialect class m	ethod),	511	
	570		find_elems_with()	(trans-
find_c	delimiter()	(trans-	late.storage.placeables.base.X	method),
	late.storage.properties.DialectFlex	class	509	
	method), 570		find_elems_with()	(trans-
find_c	delimiter()	(trans-	late.storage.placeables.general.Alt Attaches the state of the state	rPlaceable
	late.storage.properties.DialectGaia	class	method), 512	
	<i>method</i> ), 571		find_elems_with()	(trans-
find_c	delimiter()	(trans-	late.storage.placeables.general.XMLE	Entity Placeable
	late.storage.properties.DialectJava	class	method), 514	
	method), 571		<pre>find_elems_with()</pre>	(trans-
find_c	delimiter()	(trans-	late.storage.placeables.general.XMLT	agPlaceable
	late. storage. properties. Dialect Java Utf 8	class	method), 516	
	method), 572		find_elems_with()	(trans-
find_c	delimiter()	(trans-	late.storage.placeables.interfaces.Base	ePlaceable
	late.storage.properties.DialectJoomla	class	method), 518	
	method), 572		find_elems_with()	(trans-
find_c	delimiter()	(trans-	late.storage.placeables.interfaces.Invi	sible Placeable
	late.storage.properties.DialectMozilla	class	method), 519	
	method), 573		<pre>find_elems_with()</pre>	(trans-
find_c	delimiter()	(trans-	late.storage.placeables.interfaces.Mas	skingPlaceable

<i>method</i> ), 521		findid	() (translate.storage.html.htmlfile method), 435
<pre>find_elems_with()</pre>	(trans-	findid	\
late. storage. placeables. interfaces. Republication of the property of the	lacementPl		
method), 523			() (translate.storage.ical.icalfile method), 441
<pre>find_elems_with()</pre>			() (translate.storage.ini.inifile method), 446
late.storage.placeables.interfaces.Subj method), 524	flowPlacea	<i>b<b>f</b>e</i> indid	() (translate.storage.jsonl10n.I18NextFile method), 452
<pre>find_elems_with()     late.storage.placeables.strelem.Stringl</pre>	(trans-	findid	() (translate.storage.jsonl10n.JsonFile method), 457
method), 527	ziem	findid	() (translate.storage.jsonl10n.JsonNestedFile
find_elems_with()	(trans-	IIIIGIG	method), 458
	•	Ptaceable	() (translate.storage.jsonl10n.WebExtensionJsonFile
method), 529			method), 466
<pre>find_elems_with()</pre>	(trans-	findid	() (translate.storage.lisa.LISAfile method), 471
late.storage.placeables.xliff.Bpt	method),	findid	() (translate.storage.mo.mofile method), 477
531		findid	$() \qquad (translate.storage.mozilla\_lang.LangStore$
<pre>find_elems_with()</pre>	(trans-		method), 483
late.storage.placeables.xliff.Bx 536	method),	findid	() (translate.storage.omegat.OmegaTFile method), 488
<pre>find_elems_with()</pre>	(trans-	findid	$() \qquad (translate.storage.omegat.OmegaTFileTab$
0 1 1 1	method),		method), 490
532			() (translate.storage.php.phpfile method), 547
find_elems_with()	(trans-	findid	() (translate.storage.pocommon.pofile method),
late.storage.placeables.xliff.Ex 537	method),	findid	552 () (translate.storage.poxliff.PoXliffFile
<pre>find_elems_with()</pre>	(trans-		method), 560
$late.storage.placeables.xliff.G\ method)$	), 539	findid	
<pre>find_elems_with()</pre>	(trans-		method), 575
late.storage.placeables.xliff.It method)		findid	
find_elems_with()	(trans-	61 11 1	method), 577
late.storage.placeables.xliff.Ph 544	method),	findid	() (translate.storage.properties.joomlafile method), 579
<pre>find_elems_with()</pre>	(trans-	findid	$() \qquad (translate. storage. properties. propfile$
0 1	method),		method), 580
542		findid	
find_elems_with()	(trans-		method), 585
late.storage.placeables.xliff.Unknown2 method), 545	XML	findid	() (translate.storage.properties.stringsutf8file method), 587
<pre>find_elems_with()</pre>	(trans-	findid	() (translate.storage.pypo.pofile method), 589
late.storage.placeables.xliff.X method)			() (translate.storage.qm.qmfile method), 595
<pre>find_matches() (in module translate.tools</pre>	.pogrep),		() (translate.storage.qph.QphFile method), 600
689			() (translate.storage.rc.rcfile method), 606
<pre>find_placeable_dom_tree_roots()</pre>		findid	() (translate.storage.subtitles.AdvSubStationAlphaFile
ule translate.storage.xml_extract.g	generate),	C' 1' 1	method), 613
679	,	findid	
findaccelerators() (in module late.filters.decoration), 347	trans-	£ ; ~ d ; d	method), 614
	tion Store	findid	() (translate.storage.subtitles.SubRipFile method), 616
findid() (translate.storage.base.Transla method), 408	iionsiore	findid	() (translate.storage.subtitles.SubStationAlphaFile
findid() (translate.storage.catkeys.Ca	ıtkevsFile	ттпити	method), 618
method), 415	myst m	findid	
findid() (translate.storage.csvl10n.csvfile	method).		method), 619
420		findid	() (translate.storage.tbx.tbxfile method), 624
findid() (translate.storage.dtd.dtdfile method	<i>1</i> ), 426		() (translate.storage.tiki.TikiStore method), 630

findid() (translate.storage.tmx.tmxfile method), 635 findunit() (translate.storage.properties.propfile (translate.storage.trados.TradosTxtTmFilemethod), 580 findid() findunit() *method*), 644 (translate.storage.properties.stringsfile findid() (translate.storage.ts2.tsfile method), 646 method), 585 findid() (translate.storage.txt.TxtFile method), 651 findunit () (translate.storage.properties.stringsutf8file findid() (translate.storage.utx.UtxFile method), 657 method), 587 findid() (translate.storage.wordfast.WordfastTMFile (translate.storage.pypo.pofile method), findunit() 589 *method*), 667 findid() (translate.storage.xliff.xlifffile method), 673 findunit () (translate.storage.qm.qmfile method), 596 module findunit() (translate.storage.qph.QphFile method), findmarkedvariables() (in late.filters.decoration), 347 600 findunit() (translate.storage.base.TranslationStore findunit () (translate.storage.rc.rcfile method), 606 findunit () (translate.storage.subtitles.AdvSubStationAlphaFile *method*), 408 (translate.storage.catkeys.CatkeysFile findunit() *method*), 613 method), 415 findunit () (translate.storage.subtitles.MicroDVDFile findunit() (translate.storage.csvl10n.csvfile method), 614 method), 420 (translate.storage.subtitles.SubRipFile findunit() findunit () (translate.storage.dtd.dtdfile method), 426 method), 616  $\verb|findunit()| (\textit{translate.storage.subtitles.SubStationAlphaFile}|$ findunit() (translate.storage.html.htmlfile method), 435 method), 618 findunit() (translate.storage.html.POHTMLParser findunit() (translate.storage.subtitles.SubtitleFile *method*), 433 method), 619 findunit() (translate.storage.ical.icalfile method), findunit () (translate.storage.tbx.tbxfile method), 624 findunit() (translate.storage.tiki.TikiStore method), findunit () (translate.storage.ini.inifile method), 446 630 findunit() (translate.storage.jsonl10n.I18NextFile findunit() (translate.storage.tmx.tmxfile method), *method*), 452 635 (translate.storage.jsonl10n.JsonFile findunit()(translate.storage.trados.TradosTxtTmFile findunit() method), 457 method), 644 findunit () (translate.storage.jsonl10n.JsonNestedFile findunit () (translate.storage.ts2.tsfile method), 646 method), 458 findunit() (translate.storage.txt.TxtFile method), findunit () (translate.storage.jsonl10n.WebExtensionJsonFile 651 method), 466 findunit() (translate.storage.utx.UtxFile method), 657 findunit() (translate.storage.lisa.LISAfile method), 471 findunit() (translate.storage.wordfast.WordfastTMFile findunit () (translate.storage.mo.mofile method), 477 *method*), 667 findunit () (translate.storage.mozilla lang.LangStore findunit() (translate.storage.xliff.xlifffile method), *method*), 483 673 findunit() (translate.storage.omegat.OmegaTFile findunits()(translate.storage.base.TranslationStore *method*), 488 method), 408 findunit()(translate.storage.omegat.OmegaTFileTab (translate.storage.catkeys.CatkeysFile findunits() *method*), 490 method), 415 findunit() (translate.storage.php.phpfile method), findunits() (translate.storage.csvl10n.csvfile 548 method), 420 findunits() (translate.storage.dtd.dtdfile method), findunit() (translate.storage.pocommon.pofile method), 552 427 findunit() (translate.storage.poxliff.PoXliffFile findunits()(translate.storage.html.htmlfile method),

435

441

446

method), 433

findunits() (translate.storage.html.POHTMLParser

findunits() (translate.storage.ical.icalfile method),

findunits() (translate.storage.ini.inifile method),

735

*method*), 560

method), 575

*method*), 577

*method*), 579

(translate.storage.properties.javafile

(translate.storage.properties.joomlafile

findunit() (translate.storage.properties.javautf8file

findunit()

findunit()

*method*), 452 findunits() (translate.storage.jsonl10n.JsonFile method), 457 findunits() (translate.storage.jsonl10n.JsonNestedFile method), 458 findunits() (translate.storage.jsonl10n.WebExtensionJsonFile *method*), 466 findunits() (translate.storage.lisa.LISAfile method), 471 findunits() (translate.storage.mo.mofile method), 477 findunits() (translate.storage.mozilla\_lang.LangStore method), 483 findunits() (translate.storage.omegat.OmegaTFile *method*), 488 findunits() (translate.storage.omegat.OmegaTFileTab method), findunits() (translate.storage.php.phpfile method), 548 findunits() (translate.storage.pocommon.pofile method), 552 findunits() (translate.storage.poxliff.PoXliffFile method), 560 findunits() (translate.storage.properties.javafile method), 575 findunits()(translate.storage.properties.javautf8file method), 577 findunits() (translate.storage.properties.joomlafile *method*), 579 findunits() (translate.storage.properties.propfile *method*), 580 findunits() (translate.storage.properties.stringsfile method), 585 findunits() (translate.storage.properties.stringsutf8file method), findunits() (translate.storage.pypo.pofile method), findunits() (translate.storage.qm.qmfile method), findunits() (translate.storage.qph.QphFile method), 601 findunits () (translate.storage.rc.rcfile method), 606 findunits() (translate.storage.subtitles.AdvSubStationAlphaFile*method*), 613 findunits() late.storage.subtitles.MicroDVDFile method),

614

findunits() (translate.storage.jsonl10n.I18NextFile findunits() (translate.storage.subtitles.SubRipFile *method*), 616 findunits() late.storage.subtitles.SubStationAlphaFilemethod), 618 findunits() (translate.storage.subtitles.SubtitleFile method), 619 findunits() (translate.storage.tbx.tbxfile method), 624 findunits()(translate.storage.tiki.TikiStore method), findunits() (translate.storage.tmx.tmxfile method), findunits() (translate.storage.trados.TradosTxtTmFile method), 644 findunits () (translate.storage.ts2.tsfile method), 646 findunits() (translate.storage.txt.TxtFile method), 652 findunits() (translate.storage.utx.UtxFile method), 657 findunits() (translate.storage.wordfast.WordfastTMFile method), 667 findunits() (translate.storage.xliff.xlifffile method), 673 (translate.storage.placeables.base.Bpt flatten() method), 498 (translate.storage.placeables.base.Bx flatten() method), 506 flatten() (translate.storage.placeables.base.Ept method), 499 flatten() (translate.storage.placeables.base.Ex method), 507 (translate.storage.placeables.base.G flatten() method), 504 flatten() (translate.storage.placeables.base.It method), 503 (translate.storage.placeables.base.Ph flatten() method), 501 (translate.storage.placeables.base.Sub flatten() method), 511 (translate.storage.placeables.base.X flatten() method), 509  $\verb|flatten()| (translate.storage.placeables.general.AltAttrPlaceable$ method), 512 flatten() (translate.storage.placeables.general.XMLEntityPlaceable method), 514 flatten() (translate.storage.placeables.general.XMLTagPlaceable method), 516

 ${\tt flatten}$  () (translate.storage.placeables.interfaces.BasePlaceable

flatten() (translate.storage.placeables.interfaces.InvisiblePlaceable

736 Index

method), 518

method), 519

kingPlacealmlethod), 255
format_manpage() (trans-
acementPld <b>c&amp;able</b> nvert.po2wordfast.WfOptionParser method), 259
on Place able anpage () (trans-
$late. filters. pofilter. Filter Option Parser\ method),$
<i>Clem</i> 349
format_manpage() (trans-
rminologyP <b>latæntise</b> .optrecurse.RecursiveOptionParser method), 391
format_manpage() (trans-
late. tools. poconflicts. Conflict Option Parser
method), 683
format_manpage() (trans-
late.tools.pogrep.GrepOptionParser method), 687
format_manpage() (trans-
late.tools.porestructure.SplitOptionParser
method), 691
format_manpage() (trans-
late.tools.poterminology.TerminologyOptionParser method), 694
fr (class in translate.lang.fr), 367
fromkeys() (translate.misc.dictutils.cidict method),
386
fromkeys() (translate.storage.oo.unormalizechar
ML method), 496
ftl_format() (translate.filters.checks.L20nChecker
method), 300
fullname (translate.lang.common.Common attribute),
rser 358
funcmatch() (in module translate.filters.helpers), 348
umonDastabaseh () (in module translate.filters.helpers), 348
Lucene Daitabase() (trans-
late.filters.checks.CCLicenseChecker method),
nDatabase 271
functions() (translate.filters.checks.DrupalChecker
method), 277
functions() (translate.filters.checks.GnomeChecker
method), 282
functions () (translate.filters.checks.IOSChecker method), 288
functions() (translate.filters.checks.KdeChecker
r method), 294
functions() (translate.filters.checks.L20nChecker
method), 300
functions() (trans-
$late. filters. checks. Libre Of fice Checker\ method),$
306
functions() (trans-
late.filters.checks.MinimalChecker method),
311
functions() (translate.filters.checks.MozillaChecker

method), 317	method), 414
functions() (trans-	<pre>get_filename_type() (trans-</pre>
late.filters.checks.OpenOfficeChecker method), 323	late.storage.projstore.ProjectStore method), 568
<pre>functions() (trans- late.filters.checks.ReducedChecker method),</pre>	<pre>get_from_lines() (translate.tools.pydiff.FileDiffer     method), 698</pre>
328	<pre>get_ignored_filters() (trans-</pre>
<pre>functions() (trans- late.filters.checks.StandardChecker method),</pre>	late.filters.checks.CCLicenseChecker method), 271
334	<pre>get_ignored_filters() (trans-</pre>
functions () (translate.filters.checks.TermChecker method), 341	late.filters.checks.DrupalChecker method), 277
fuzzy_unitcount() (trans-	<pre>get_ignored_filters() (trans-</pre>
late.storage.statistics.Statistics method), 611	late.filters.checks.GnomeChecker method), 283
fuzzy_units() (translate.storage.statistics.Statistics method), 611	get_ignored_filters() (trans- late.filters.checks.IOSChecker method), 288
0	<pre>get_ignored_filters() (trans-</pre>
<b>G</b> (class in translate.storage.placeables.base), 503	late.filters.checks.KdeChecker method), 294
G (class in translate.storage.placeables.xliff), 538	<pre>get_ignored_filters() (trans-</pre>
gconf() (translate.filters.checks.GnomeChecker method), 283	late.filters.checks.L20nChecker method), 300
GenericRevisionControlSystem (class in trans-	<pre>get_ignored_filters() (trans-</pre>
late.storage.versioncontrol), 662 get () (translate.misc.dictutils.cidict method), 386	late.filters.checks.LibreOfficeChecker method), 306
get() (translate.storage.oo.unormalizechar method),	<pre>get_ignored_filters() (trans-</pre>
496	late.filters.checks.MinimalChecker method), 311
get_abs_data_filename() (in module trans-	get_ignored_filters() (trans-
late.misc.file_discovery), 387 get_all_languages() (in module trans-	late.filters.checks.MozillaChecker method), 317
<pre>late.lang.factory), 365 get_available_version_control_systems()</pre>	get_ignored_filters() (trans-
(in module translate.storage.versioncontrol), 662	late,filters.checks.OpenOfficeChecker method), 323
get_country_iso_name() (in module trans-	<pre>get_ignored_filters() (trans-</pre>
late.lang.data), 360	late.filters.checks.ReducedChecker method), 329
<pre>get_field_analyzers() (trans- late.search.indexing.CommonIndexer.CommonDe</pre>	
method), 398	late.filters.checks.StandardChecker method),
get_field_analyzers() (trans-	334
late.search.indexing.PyLuceneIndexer.PyLucene method), 401	
get_field_analyzers() (trans-	method), 338
late.search.indexing.XapianIndexer.XapianDatal method), 403	late.filters.checks.TermChecker method),
<pre>get_file() (translate.storage.bundleprojstore.BundleP</pre>	rojectStore <sup>341</sup>
method), 413	get_ignored_filters() (trans-
<pre>get_file() (translate.storage.project.Project     method), 567</pre>	late.filters.checks.TranslationChecker method), 345
<pre>get_file() (translate.storage.projstore.ProjectStore</pre>	<pre>get_ignored_filters() (trans-</pre>
method), 568	late.filters.checks.UnitChecker method),
<pre>get_filename_type() (trans- late.storage.bundleprojstore.BundleProjectStore</pre>	346 get_index_data() (trans-

late.storage.placeables.base.Bpt 498	method),	late.storage.placeables.terminology.TerminologyPlaceable method), 529
<pre>get_index_data()</pre>	(trans-	<pre>get_index_data() (trans-</pre>
late.storage.placeables.base.Bx 506	method),	late.storage.placeables.xliff.Bpt method), 531
<pre>get_index_data()</pre>	(trans-	<pre>get_index_data() (trans-</pre>
late.storage.placeables.base.Ept 499	method),	late.storage.placeables.xliff.Bx method), 536
get_index_data()	(trans-	<pre>get_index_data() (trans-</pre>
late.storage.placeables.base.Ex 507	method),	late.storage.placeables.xliff.Ept method), 532
<pre>get_index_data()</pre>	(trans-	<pre>get_index_data() (trans-</pre>
late.storage.placeables.base.G 504	method),	late.storage.placeables.xliff.Ex method), 537
<pre>get_index_data()</pre>	(trans-	<pre>get_index_data() (trans-</pre>
late.storage.placeables.base.It	method),	late.storage.placeables.xliff.G method), 539
503		<pre>get_index_data() (trans-</pre>
<pre>get_index_data()</pre>	(trans-	late.storage.placeables.xliff.It method), 541
late.storage.placeables.base.Ph	method),	<pre>get_index_data() (trans-</pre>
501		late.storage.placeables.xliff.Ph method),
<pre>get_index_data()</pre>	(trans-	544
late.storage.placeables.base.Sub	method),	<pre>get_index_data() (trans-</pre>
511		late.storage.placeables.xliff.Sub method),
<pre>get_index_data()</pre>	(trans-	542
late.storage.placeables.base.X	method),	<pre>get_index_data() (trans-</pre>
509		late. storage. placeables. x liff. UnknownXML
<pre>get_index_data()</pre>	(trans-	method), 545
late.storage.placeables.general.Alt	AttrPlaceable	
method), 513		$late.storage.placeables.xliff.X\ method), 534$
<pre>get_index_data()     late.storage.placeables.general.XM</pre>	trans- LEntityPlace	get_language_iso_fullname() (in module able translate.lang.data), 360
method), 514		<pre>get_language_iso_name() (in module trans-</pre>
<pre>get_index_data()</pre>	(trans-	late.lang.data), 360
late.storage.placeables.general.XM	ILTagPlaceab	pleget_matches() (trans-
method), 516		late. search. in dexing. Common Indexer. Common Enquire
get_index_data()	(trans-	method), 399
late.storage.placeables.interfaces.E	BasePlaceable	e get_matches() (trans-
method), 518		late.search.indexing.PyLuceneIndexer.PyLuceneHits
<pre>get_index_data()</pre>	(trans-	method), $402$
late.storage.placeables.interfaces.I	nvisiblePlace	_ ·
method), 519		late.search.indexing.XapianIndexer.XapianEnquire
<pre>get_index_data()</pre>	(trans-	method), 404
late.storage.placeables.interfaces.M	<i>1</i> askingPlace	
method), 521	,	late.search.indexing.CommonIndexer.CommonEnquire
<pre>get_index_data()</pre>	(trans-	method), 399
late.storage.placeables.interfaces.F	ReplacementP	
method), 523	,	late.search.indexing.PyLuceneIndexer.PyLuceneHits
<pre>get_index_data()</pre>	(trans-	method), 402
late.storage.placeables.interfaces.S	ubflowPlaced	
method), 525		late.search.indexing.XapianIndexer.XapianEnquire
get_index_data()	(trans-	method), 405
late.storage.placeables.strelem.Stri	пдЕлет	get_parent_elem() (trans-
method), 527	(+	late.storage.placeables.base.Bpt method),
<pre>get_index_data()</pre>	(trans-	498

<pre>get_parent_elem()</pre>	(trans-	<pre>get_parent_elem()</pre>	(trans-
late.storage.placeables.base.Bx 506	method),	late.storage.placeables.xliff.Bpt 531	method),
<pre>get_parent_elem()</pre>	(trans-	<pre>get_parent_elem()</pre>	(trans-
late.storage.placeables.base.Ept 500	method),	late.storage.placeables.xliff.Bx 536	method),
<pre>get_parent_elem()</pre>	(trans-	<pre>get_parent_elem()</pre>	(trans-
late.storage.placeables.base.Ex 508	method),	late.storage.placeables.xliff.Ept 533	method),
<pre>get_parent_elem()</pre>	(trans-	<pre>get_parent_elem()</pre>	(trans-
late.storage.placeables.base.G 504	method),	late.storage.placeables.xliff.Ex 537	method),
<pre>get_parent_elem()</pre>	(trans-	<pre>get_parent_elem()</pre>	(trans-
late. storage. placeables. base. It	method),	late.storage.placeables.xliff.G meth	od), 539
503		<pre>get_parent_elem()</pre>	(trans-
<pre>get_parent_elem()</pre>	(trans-	late.storage.placeables.xliff.It metho	
late.storage.placeables.base.Ph	method),	<pre>get_parent_elem()</pre>	(trans-
501		late.storage.placeables.xliff.Ph	method),
<pre>get_parent_elem()</pre>	(trans-	544	
late.storage.placeables.base.Sub	method),	<pre>get_parent_elem()</pre>	(trans-
511	( )	late.storage.placeables.xliff.Sub	method),
get_parent_elem()	(trans-	542	(tuana
late.storage.placeables.base.X 509	method),	get_parent_elem()	(trans-
get_parent_elem()	(trans-	late.storage.placeables.xliff.Unknov method), 546	VIIAWIL
late.storage.placeables.general.Alt.	•		(trans-
method), 513		$late.storage.placeables.xliff.X\ methods$	`
<pre>get_parent_elem()</pre>	(trans-	<pre>get_proj_filename()</pre>	(trans-
late.storage.placeables.general.XM method), 514		method), 414	ProjectStore
<pre>get_parent_elem()</pre>		<pre>get_proj_filename()</pre>	(trans-
late.storage.placeables.general.XM	ILTagPlaceabl		), 567
method), 516		<pre>get_proj_filename()</pre>	(trans-
<pre>get_parent_elem()</pre>	(trans-	late.storage.projstore.ProjectStore	method),
late.storage.placeables.interfaces.h	<i>BasePlaceable</i>		(1
method), 518	(tuana	get_query_result()	(trans-
<pre>get_parent_elem()     late.storage.placeables.interfaces.I</pre>	(trans-	late.search.indexing.CommonIndexeable method), 398	er.CommonDaiabase
method), 520	nvisibiei iuced	get_query_result()	(trans-
get_parent_elem()	(trans-	late.search.indexing.PyLuceneIndex	,
late.storage.placeables.interfaces.M	`		ст. увисстевинавизе
method), 521	raskingi tacce	get_query_result()	(trans-
<pre>get_parent_elem()</pre>	(trans-	late.search.indexing.XapianIndexer.	,
late.storage.placeables.interfaces.I	•		1
method), 523		<pre>get_real_filename()</pre>	(trans-
<pre>get_parent_elem()</pre>	(trans-	late.storage.project.Project method)	), 567
late.storage.placeables.interfaces.S	SubflowPlacea	<i>bb</i> et_rich_target()	(trans-
method), 525		late. storage. pox liff. PoX liff Unit	method),
<pre>get_parent_elem()</pre>	(trans-	564	
late.storage.placeables.strelem.Stri	ingElem	<pre>get_rich_target() (translate.storage</pre>	xliff.xliffunit
method), 527	,	method), 676	
<pre>get_parent_elem()</pre>		get_source_text()	(trans-
late.storage.placeables.terminolog method), 529	y. Ierminology	Placeable late.storage.statistics.Statistics 611	method),
•			

<pre>get_starttag_text() (trans-</pre>	getcleanfile() (trans-
late.storage.html.htmlfile method), 435	late.storage.versioncontrol.bzr.bzr method),
<pre>get_starttag_text() (trans-</pre>	663
late.storage.html.POHTMLParser method),	getcleanfile() (trans-
433	late.storage.versioncontrol.cvs.cvs method),
<pre>get_time() (translate.storage.trados.TradosTxtDate</pre>	663
method), 640	getcleanfile() (trans-
<pre>get_time() (translate.storage.wordfast.WordfastTime</pre>	late.storage.versioncontrol.darcs.darcs
method), 668	method), 664
<pre>get_timestring() (trans-</pre>	getcleanfile() (trans-
late.storage.trados.TradosTxtDate method), 640	late.storage.versioncontrol.GenericRevisionControlSystem method), 662
<pre>get_timestring() (trans-</pre>	<pre>getcleanfile() (trans-</pre>
late.storage.wordfast.WordfastTime method), 668	late.storage.versioncontrol.git.git method), 664
<pre>get_to_lines() (translate.tools.pydiff.FileDiffer</pre>	getcleanfile() (trans-
method), 698	late.storage.versioncontrol.hg.hg method),
<pre>get_version() (in module trans-</pre>	665
late.storage.versioncontrol.bzr), 663	getcleanfile() (trans-
get_version() (in module trans-	late.storage.versioncontrol.svn.svn method),
late.storage.versioncontrol.hg), 664	665
<pre>get_version() (in module trans-</pre>	getcontext() (trans-
late.storage.versioncontrol.svn), 665	late.storage.base.TranslationUnit method),
<pre>get_versioned_object() (in module trans-</pre>	411
late.storage.versioncontrol), 662	getcontext() (translate.storage.catkeys.CatkeysUnit
<pre>get_versioned_objects_recursive() (in</pre>	method), 417
module translate.storage.versioncontrol), 663	<pre>getcontext() (translate.storage.csvl10n.csvunit</pre>
<pre>get_xliff_source_target_doms() (in module</pre>	method), 422
translate.storage.xml_extract.generate), 679	getcontext() (translate.storage.dtd.dtdunit method),
getaccelerators() (in module trans-	428
late.filters.decoration), 347	getcontext() (translate.storage.html.htmlunit
<pre>getalttrans() (translate.storage.poxliff.PoXliffUnit</pre>	method), 438
method), 564	getcontext() (translate.storage.ical.icalunit
getalttrans() (translate.storage.pypo.pounit	method), 443
method), 592	getcontext() (translate.storage.ini.iniunit method),
getalttrans() (translate.storage.xliff.xliffunit	448
method), 676	getcontext() (trans-
J	late.storage.jsonl10n.I18NextUnit method),
late.convert.convert.ArchiveConvertOptionParser	
method), 234	getcontext() (trans-
getarchiveclass() (trans-	late.storage.jsonl10n.JsonNestedUnit method),
late.convert.po2tmx.TmxOptionParser	460
method), 255	getcontext() (translate.storage.jsonl10n.JsonUnit
getarchiveclass() (trans-	method), 463
late.convert.po2wordfast.WfOptionParser	getcontext() (trans-
method), 260	late.storage.jsonl10n.WebExtensionJsonUnit
getautomaticcomments() (trans-	method), 468
late.storage.poxliff.PoXliffUnit method), 564	getcontext() (translate.storage.lisa.LISAunit method), 473
getbodynode() (translate.storage.poxliff.PoXliffFile	getcontext() (translate.storage.mo.mounit method),
method), 560	480
getbodynode() (translate.storage.xliff.xlifffile	getcontext() (trans-
method), 673	late.storage.mozilla_lang.LangUnit method),
<pre>getclass() (in module translate.storage.factory), 431</pre>	485

(transmethod), 417 getcontext() late.storage.omegat.OmegaTUnit (translate.storage.omegat.OmegaTUnit method), getdict() 492 method), 492 (translate.storage.php.phpunit (translate.storage.utx.UtxUnit method), getcontext() getdict() method), 549 659 (translate.storage.wordfast.WordfastUnit getcontext() (translate.storage.pocommon.pounit getdict() method), 555 method), 669 getcontext() (translate.storage.poxliff.PoXliffUnit getElementsByTagName\_helper() (in module method), 564 translate.misc.ourdom), 393 getcontext() (translate.storage.properties.propunit getemails() (in module translate.filters.decoration), *method*), 582 347 geterrors() (translate.storage.base.TranslationUnit getcontext() (translate.storage.pypo.pounit method), 592 method), 411 getcontext() (translate.storage.qm.qmunit method), geterrors() (translate.storage.catkeys.CatkeysUnit 597 method), 417 getcontext() (translate.storage.qph.QphUnit (translate.storage.csvl10n.csvunit geterrors() method), 603 method), 422 getcontext() (translate.storage.rc.rcunit method), geterrors() (translate.storage.dtd.dtdunit method), 608 428 getcontext() (transgeterrors() (translate.storage.html.htmlunit late.storage.subtitles.SubtitleUnit method), method), 438 geterrors() (translate.storage.ical.icalunit method), getcontext() (translate.storage.tbx.tbxunit method), 443 geterrors() (translate.storage.ini.iniunit method), 626 getcontext() (translate.storage.tiki.TikiUnit 448 *method*), 632 geterrors() (translate.storage.jsonl10n.118NextUnit (translate.storage.tmx.tmxunit getcontext() method), 454 method), 637 geterrors() (trans-(translate.storage.trados.TradosUnit late.storage.jsonl10n.JsonNestedUnit method), getcontext() 460 method), 641 getcontext() (translate.storage.ts2.tsunit method), geterrors() (translate.storage.jsonl10n.JsonUnit method), 464 getcontext() (translate.storage.txt.TxtUnit method), (transgeterrors() 653 late. storage. js on 110n. Web Extension Js on Unit(translate.storage.utx.UtxUnit method), 468 getcontext() (translate.storage.lisa.LISAunit *method*), 659 geterrors() getcontext() (trans*method*), 473 late.storage.wordfast.WordfastUnit geterrors() (translate.storage.mo.mounit method), *method*), 480 getcontext() (translate.storage.xliff.xliffunit geterrors() (translate.storage.mozilla lang.LangUnit *method*), 676 method), getcontextgroups() (translate.storage.poxliff.PoXliffUnit method), geterrors() (translate.storage.omegat.OmegaTUnit 564 method), 492 getcontextgroups() (transgeterrors() (translate.storage.php.phpunit method), late.storage.xliff.xliffunit method), 676 549 getdatatype() (translate.storage.poxliff.PoXliffFile geterrors() (translate.storage.pocommon.pounit *method*), 560 method), 555 getdatatype() (translate.storage.xliff.xlifffile (translate.storage.poxliff.PoXliffUnit geterrors() method), 673 method), 564 (translate.storage.poxliff.PoXliffFile geterrors() (translate.storage.properties.propunit getdate() method), 561 method), 582 getdate() (translate.storage.xliff.xlifffile method), 673 geterrors() (translate.storage.pypo.pounit method), getdict() (translate.storage.catkeys.CatkeysUnit 592

<pre>geterrors() (translate.storage.qm.qmunit method),</pre>	method), 294
597 geterrors() (translate.storage.qph.QphUnit	<pre>getfilters() (translate.filters.checks.L20nChecker</pre>
method), 603	getfilters() (trans-
geterrors() (translate.storage.rc.rcunit method), 608 geterrors() (translate.storage.subtitles.SubtitleUnit	late.filters.checks.LibreOfficeChecker method), 306
method), 621	getfilters() (trans-
geterrors() (translate.storage.tbx.tbxunit method), 626	late.filters.checks.MinimalChecker method), 311
geterrors() (translate.storage.tiki.TikiUnit method), 632	getfilters() (trans- late.filters.checks.MozillaChecker method),
<pre>geterrors() (translate.storage.tmx.tmxunit method),</pre>	317
637	getfilters() (trans-
geterrors() (translate.storage.trados.TradosUnit method), 641	late.filters.checks.OpenOfficeChecker method), 323
<pre>geterrors() (translate.storage.ts2.tsunit method),</pre>	getfilters() (trans-
648 geterrors() (translate.storage.txt.TxtUnit method),	late.filters.checks.ReducedChecker method), 329
653	getfilters() (trans-
geterrors() (translate.storage.utx.UtxUnit method), 659	late.filters.checks.StandardChecker method), 334
geterrors() (trans- late.storage.wordfast.WordfastUnit method), 669	getfilters() (trans- late.filters.checks.StandardUnitChecker method), 338
geterrors() (translate.storage.xliff.xliffunit method), 676	getfilters() (translate.filters.checks.TeeChecker method), 339
getfilename() (translate.storage.poxliff.PoXliffFile method), 561	getfilters() (translate.filters.checks.TermChecker method), 342
getfilename() (translate.storage.xliff.xlifffile	getfilters() (trans-
method), 673	late.filters.checks.TranslationChecker method), 345
<pre>getfilenames() (translate.storage.poxliff.PoXliffFile</pre>	<pre>getfilters() (translate.filters.checks.UnitChecker</pre>
getfilenames() (translate.storage.xliff.xlifffile	method), 346
method), 673	getformathelp() (trans-
<pre>getfilenode() (translate.storage.poxliff.PoXliffFile</pre>	late.convert.convert.ArchiveConvertOptionParse method), 234
getfilenode() (translate.storage.xliff.xlifffile	getformathelp() (trans-
method), 673	late.convert.convert.ConvertOptionParser
getfiles() (translate.storage.directory.Directory	method), 238
method), 425	getformathelp() (trans-
getfiles() (translate.storage.zip.ZIPFile method), 681	late.convert.po2moz.MozConvertOptionParser method), 250
getfilters() (trans-	getformathelp() (trans-
late.filters.checks.CCLicenseChecker method), 271	late.convert.po2tmx.TmxOptionParser method), 255
getfilters() (trans-	getformathelp() (trans-
late.filters.checks.DrupalChecker method), 277	late.convert.po2wordfast.WfOptionParser method), 260
<pre>getfilters() (trans-</pre>	getformathelp() (trans-
late.filters.checks.GnomeChecker method),	late.filters.pofilter.FilterOptionParser method), 349
getfilters() (translate.filters.checks.IOSChecker	
method), 288	late.misc.opt recurse. Recursive Option Parser
getfilters() (translate.filters.checks.KdeChecker	method), 391

<pre>getformathelp() (trans- late.tools.poconflicts.ConflictOptionParser method), 683</pre>	<pre>getfulloutputpath() (trans- late.convert.po2tmx.TmxOptionParser     method), 255</pre>
getformathelp() (trans-	getfulloutputpath() (trans-
late.tools.pogrep.GrepOptionParser method), 687	late.convert.po2wordfast.WfOptionParser method), 260
getformathelp() (trans-	getfulloutputpath() (trans-
late.tools.porestructure.SplitOptionParser method), 691	late.filters.pofilter.FilterOptionParser method), 349
	getfulloutputpath() (trans-
late.tools.poterminology.TerminologyOptionPars method), 694	er late.misc.optrecurse.RecursiveOptionParser method), 391
getfullinputpath() (trans-	
late.convert.convert.ArchiveConvertOptionParset method), 234	r late.tools.poconflicts.ConflictOptionParser method), 684
getfullinputpath() (trans-	getfulloutputpath() (trans-
late.convert.convert.ConvertOptionParser method), 238	late.tools.pogrep.GrepOptionParser method), 687
getfullinputpath() (trans-	getfulloutputpath() (trans-
late.convert.po2moz.MozConvertOptionParser method), 250	late.tools.porestructure.SplitOptionParser method), 691
getfullinputpath() (trans-	getfulloutputpath() (trans-
late.convert.po2tmx.TmxOptionParser method), 255	late.tools.poterminology.TerminologyOptionParser method), 695
getfullinputpath() (trans-	getfulltemplatepath() (trans-
late.convert.po2wordfast.WfOptionParser method), 260	late.convert.convert.ArchiveConvertOptionParser method), 234
getfullinputpath() (trans-	getfulltemplatepath() (trans-
late.filters.pofilter.FilterOptionParser method), 349	late.convert.convert.ConvertOptionParser method), 238
getfullinputpath() (trans-	<pre>getfulltemplatepath() (trans-</pre>
late.misc.optrecurse.RecursiveOptionParser method), 391	late.convert.po2moz.MozConvertOptionParser method), 250
getfullinputpath() (trans-	getfulltemplatepath() (trans-
late.tools.poconflicts.ConflictOptionParser method), 684	late.convert.po2tmx.TmxOptionParser method), 255
getfullinputpath() (trans-	J 1 1 1
late.tools.pogrep.GrepOptionParser method), 687	late.convert.po2wordfast.WfOptionParser method), 260
getfullinputpath() (trans-	getfulltemplatepath() (trans-
late.tools.porestructure.SplitOptionParser method), 691	late.filters.pofilter.FilterOptionParser method), 349
getfullinputpath() (trans- late.tools.poterminology.TerminologyOptionPars	
method), 694	method), 391
getfulloutputpath() (trans- late.convert.convert.ArchiveConvertOptionParset method), 234	getfulltemplatepath() (trans- r late.tools.poconflicts.ConflictOptionParser method), 684
getfulloutputpath() (trans-	getfulltemplatepath() (trans-
late.convert.convert.ConvertOptionParser method), 238	late.tools.pogrep.GrepOptionParser method), 687
getfulloutputpath() (trans-	getfulltemplatepath() (trans-
late.convert.po2moz.MozConvertOptionParser	late.tools.porestructure.SplitOptionParser
method), 250	method), 691

	getid() (translate.storage.pypo.pounit method), 592
late. tools. poterminology. Terminology Option Pars	egetid() (translate.storage.qm.qmunit method), 598
method), 695	getid() (translate.storage.qph.QphUnit method), 603
getfunctions() (in module trans-	getid() (translate.storage.rc.rcunit method), 608
late.filters.decoration), 347	getid() (translate.storage.subtitles.SubtitleUnit
getheader() (trans-	method), 621
late.storage.wordfast.WordfastHeader method),	getid() (translate.storage.tbx.tbxunit method), 626
666	getid() (translate.storage.tiki.TikiUnit method), 632
getheadernode() (trans-	getid() (translate.storage.tmx.tmxunit method), 637
late.storage.poxliff.PoXliffFile method), 561	getid() (translate.storage.trados.TradosUnit method),
getheadernode() (translate.storage.xliff.xlifffile	641
method), 673	getid() (translate.storage.ts2.tsunit method), 648
	- · · · · · · · · · · · · · · · · · · ·
getheaderplural() (translate.storage.mo.mofile	getid() (translate.storage.txt.TxtUnit method), 654
method), 477	getid() (translate.storage.utx.UtxUnit method), 659
getheaderplural() (trans-	getid() (translate.storage.wordfast.WordfastUnit
late.storage.pocommon.pofile method), 552	method), 669
getheaderplural() (trans-	getid() (translate.storage.xliff.xliffunit method), 676
late.storage.poheader.poheader method), 558	getids() (translate.storage.base.TranslationStore method), 408
<pre>getheaderplural() (trans-</pre>	getids() (translate.storage.catkeys.CatkeysFile
late.storage.poxliff.PoXliffFile method), 561	method), 415
getheaderplural() (translate.storage.pypo.pofile	getids() (translate.storage.csvl10n.csvfile method),
method), 589	420
getid() (translate.storage.base.TranslationUnit	getids() (translate.storage.dtd.dtdfile method), 427
method), 411	getids() (translate.storage.html.htmlfile method), 436
getid() (translate.storage.catkeys.CatkeysUnit	getids() (translate.storage.html.POHTMLParser
method), 417	method), 433
<pre>getid() (translate.storage.csvl10n.csvunit method),</pre>	getids() (translate.storage.ical.icalfile method), 442
422	getids() (translate.storage.ini.inifile method), 446
getid() (translate.storage.dtd.dtdunit method), 428	getids() (translate.storage.jsonl10n.118NextFile
getid() (translate.storage.html.htmlunit method), 438	method), 452
getid() (translate.storage.ical.icalunit method), 443	getids() (translate.storage.jsonl10n.JsonFile
getid() (translate.storage.ini.iniunit method), 448	method), 457
<pre>getid() (translate.storage.jsonl10n.118NextUnit</pre>	<pre>getids() (translate.storage.jsonl10n.JsonNestedFile</pre>
method), 454	method), 459
<pre>getid() (translate.storage.jsonl10n.JsonNestedUnit</pre>	<pre>getids() (translate.storage.jsonl10n.WebExtensionJsonFile</pre>
method), 460	method), 466
<pre>getid() (translate.storage.jsonl10n.JsonUnit method),</pre>	getids() (translate.storage.lisa.LISAfile method), 471
464	getids() (translate.storage.mo.mofile method), 477
<pre>getid() (translate.storage.jsonl10n.WebExtensionJsonU</pre>	
method), 468	method), 483
getid() (translate.storage.lisa.LISAunit method), 474	getids() (translate.storage.omegat.OmegaTFile
getid() (translate.storage.mo.mounit method), 480	method), 488
getid() (translate.storage.mozilla_lang.LangUnit	getids() (translate.storage.omegat.OmegaTFileTab
method), 485	method), 490
getid() (translate.storage.omegat.OmegaTUnit	getids() (translate.storage.php.phpfile method), 548
method), 492	getids()(translate.storage.pocommon.pofile method),
getid() (translate.storage.php.phpunit method), 550	552
<pre>getid() (translate.storage.pocommon.pounit method),</pre>	getids() (translate.storage.poxliff.PoXliffFile
555	method), 561
<pre>getid() (translate.storage.poxliff.PoXliffUnit method),</pre>	getids() (translate.storage.properties.javafile
564	method), 575
<pre>getid() (translate.storage.properties.propunit</pre>	getids() (translate.storage.properties.javautf8file
method) 582	method) 577

getids()	(translate.storage.properties.joomlafile	e late.storage.poxliff.PoXliffUnit 564	method),
getids()	(translate.storage.properties.propfile	e getlanguageNodes()	(trans-
	(hod), 580	late.storage.qph.QphUnit method), 60	
getids() <i>met</i>	(translate.storage.properties.stringsfile thod), 585	e getlanguageNodes() (translate.storage.tl method), 627	bx.tbxunit
getids()	(translate.storage.properties.stringsutf8file	e getlanguageNodes()	(trans-
	thod), 587	late.storage.tmx.tmxunit method), 637	`
	translate.storage.pypo.pofile method), 589	getlanguageNodes() (translate.storage.	ts2 tsunit
	translate.storage.qm.qmfile method), 596	method), 648	
	translate.storage.qph.QphFile method), 601		(trans-
	translate.storage.rc.rcfile method), 606	late.storage.xliff.xliffunit method), 676	
	translate.storage.subtitles.AdvSubStationAl		(trans-
	(hod), 613	late.storage.base.Translation Unit	method),
	(translate.storage.subtitles.MicroDVDFile	e 411	
met	(hod), 615	getlocations()	(trans-
getids()	(translate.storage.subtitles.SubRipFile	e late.storage.catkeys.CatkeysUnit 417	method),
	translate.storage.subtitles.SubStationAlpha		n csvunit
	thod), 618	method), 422	m.csvanu
	**		4.1.14.1
getids()	(translate.storage.subtitles.SubtitleFile		іа.атаипіт
	(hod), 620	method), 429	
	translate.storage.tbx.tbxfile method), 624	getlocations() (translate.storage.html	l.htmlunit
getids()(	translate.storage.tiki.TikiStore method), 630	0 <i>method</i> ), 438	
getids()(	translate.storage.tmx.tmxfile method), 635	getlocations() (translate.storage.ica	ıl.icalunit
getids()	(translate.storage.trados.TradosTxtTmFilesting) = (translate.storage.tradosTxtTmFilesting) = (translate.s	e method), 444	
	(hod), 644	getlocations() (translate.storage.i	ini.iniunit
	translate.storage.ts2.tsfile method), 646	method), 448	
	translate.storage.txt.TxtFile method), 652	getlocations()	(trans-
	translate.storage.utx.UtxFile method), 657	late.storage.jsonl10n.I18NextUnit	method),
	_		memou),
	(translate.storage.wordfast.WordfastTMFile		(1
	(hod), 667	getlocations()	(trans-
	translate.storage.xliff.xlifffile method), 673	late. storage. js on l10 n. Js on Nested Unit	method),
	translate.storage.oo.ooline method), 495	461	
getlangua	ge() (in module translate.lang.factory)	), getlocations()	(trans-
365	I	late.storage.jsonl 10 n. Json Unit	method),
getlangua	geNode () (translate.storage.lisa.LISAuni	it 464	
met	(hod), 474	getlocations()	(trans-
getlangua			onÙnit
	e.storage.poxliff.PoXliffUnit method)		
564		getlocations() (translate.storage.lisa	I ISAunit
		-	.L15/14/111
	geNode() (translate.storage.qph.QphUni		•,
	(hod), 603	getlocations() (translate.storage.m	10.mounit
getlangua	_		
	thod), 627	getlocations()	(trans-
getlangua <i>met</i>	geNode() (translate.storage.tmx.tmxuni thod), 637	it late.storage.mozilla_lang.LangUnit 485	method),
getlangua			(trans-
	(translate.storage.is2.isuni (thod), 648		method),
mei getlangua		<u> </u>	memou),
			n nhaaii
	(thod), 676	getlocations() (translate.storage.ph	р.рприпіт
getlangua		**	
	e.storage.lisa.LISAunit method), 474	getlocations()	(trans-
getlangua	geNodes() (trans	s- late.storage.pocommon.pounit	method),

555 getlocations()	(trans-	getnotes() (translate.storage.base.TranslationUnit method), 411
late.storage.po	exliff.PoXliffUnit method),	getnotes() (translate.storage.catkeys.CatkeysUnit
564 getlocations()	(trans-	method), 418 getnotes() (translate.storage.csvl10n.csvunit
582	operties.propunit method),	<pre>method), 423 getnotes() (translate.storage.dtd.dtdunit method),</pre>
getlocations() <i>method</i> ), 592	(translate.storage.pypo.pounit	429 getnotes() (translate.storage.html.htmlunit method),
<pre>getlocations()     method), 598</pre>	(translate.storage.qm.qmunit	438 getnotes() (translate.storage.ical.icalunit method),
getlocations() <i>method</i> ), 603	(translate.storage.qph.QphUnit	444 getnotes() (translate.storage.ini.iniunit method), 449
getlocations()  method), 608	(translate.storage.rc.rcunit	getnotes() (translate.storage.jsonl10n.I18NextUnit method), 454
getlocations()	trans-btitles.SubtitleUnit method),	getnotes () (translate.storage.jsonl10n.JsonNestedUnit method), 461
622 getlocations()	(translate.storage.tbx.tbxunit	getnotes() (translate.storage.jsonl10n.JsonUnit method), 464
method), 627 getlocations()	(translate.storage.tiki.TikiUnit	getnotes() (translate.storage.jsonl10n.WebExtensionJsonUnit method), 469
method), 632 getlocations()	(translate.storage.tmx.tmxunit	getnotes() (translate.storage.lisa.LISAunit method), 474
method), 637	(transtate.storage.tmx.tmx	getnotes() (translate.storage.mo.mounit method), 480
getlocations()  late.storage.tro 641	ados.TradosUnit method),	getnotes() (translate.storage.mozilla_lang.LangUnit method), 485
getlocations()  method), 648	(translate.storage.ts2.tsunit	getnotes() (translate.storage.omegat.OmegaTUnit method), 492
getlocations()  method), 654	(translate.storage.txt.TxtUnit	getnotes() (translate.storage.php.phpunit method), 550
getlocations()  method), 659	(translate.storage.utx.UtxUnit	getnotes() (translate.storage.pocommon.pounit method), 556
getlocations()	(trans-	getnotes() (translate.storage.poxliff.PoXliffUnit
670	ordfast.WordfastUnit method),	method), 565 getnotes() (translate.storage.properties.propunit
getlocations() method), 676	(translate.storage.xliff.xliffunit	method), 583 getnotes() (translate.storage.pypo.pounit method),
393	module translate.misc.ourdom),	592 getnotes() (translate.storage.qm.qmunit method),
<pre>getNodeText()     method), 473</pre>	(translate.storage.lisa.LISAunit	598 getnotes() (translate.storage.qph.QphUnit method),
getNodeText() (tra method), 564	nslate.storage.poxliff.PoXliffUnit	603 getnotes() (translate.storage.rc.rcunit method), 608
<pre>getNodeText()     method), 603</pre>	(translate.storage.qph.QphUnit	getnotes() (translate.storage.subtitles.SubtitleUnit method), 622
<pre>getNodeText()     method), 626</pre>	(translate.storage.tbx.tbxunit	getnotes() (translate.storage.tbx.tbxunit method), 627
<pre>getNodeText()     method), 637</pre>	(translate.storage.tmx.tmxunit	getnotes() (translate.storage.tiki.TikiUnit method), 632
getNodeText() (tran	nslate.storage.ts2.tsunit method),	getnotes() (translate.storage.tmx.tmxunit method), 637
<pre>getNodeText()     method), 676</pre>	(translate.storage.xliff.xliffunit	getnotes() (translate.storage.trados.TradosUnit method), 641

getnotes() (translate.storage.ts2.tsunit method), 648	687
<pre>getnotes() (translate.storage.txt.TxtUnit method),</pre>	
654	late.tools.porestructure.SplitOptionParser
getnotes() (translate.storage.utx.UtxUnit method), 659	method), 691 getoutputname() (trans-
getnotes() (translate.storage.wordfast.WordfastUnit method), 670	late.tools.poterminology.TerminologyOptionParse method), 695
getnotes() (translate.storage.xliff.xliffunit method),	
676 getnumbers() (in module trans-	late.convert.convert.ArchiveConvertOptionParser method), 234
late.filters.decoration), 347	getoutputoptions() (trans-
getobject() (in module translate.storage.factory), 432	late.convert.convert.ConvertOptionParser method), 238
getoofile() (translate.storage.oo.oomultifile	getoutputoptions() (trans-
method), 495 getoutput() (translate.storage.dtd.dtdunit method),	late.convert.po2moz.MozConvertOptionParser method), 250
429	getoutputoptions() (trans-
getoutput() (translate.storage.oo.oofile method), 495	late.convert.po2tmx.TmxOptionParser method), 255
getoutput() (translate.storage.oo.ooline method),	
495 getoutput() (translate.storage.oo.oounit method),	late.convert.po2wordfast.WfOptionParser method), 260
496	getoutputoptions() (trans-
getoutput() (translate.storage.php.phpunit method), 550	late.filters.pofilter.FilterOptionParser method), 349
getoutput() (translate.storage.properties.propunit method), 583	late.misc. optrecurse. Recursive Option Parser
getoutput () (translate.storage.rc.rcunit method), 608	method), 391
getoutputname() (trans- late.convert.convert.ArchiveConvertOptionParser method), 234	getoutputoptions() (trans- r late.tools.poconflicts.ConflictOptionParser method), 684
getoutputname() (trans-	getoutputoptions() (trans-
late.convert.convert.ConvertOptionParser method), 238	late.tools.pogrep.GrepOptionParser method), 688
getoutputname() (trans- late.convert.po2moz.MozConvertOptionParser method), 250	getoutputoptions() (trans- late.tools.porestructure.SplitOptionParser method), 691
	getoutputoptions() (trans-
late.convert.po2tmx.TmxOptionParser method), 255	late.tools.poterminology.TerminologyOptionParse method), 695
getoutputname() (trans-	getparts() (translate.storage.oo.ooline method), 495
late.convert.po2wordfast.WfOptionParser method), 260	getpassthroughoptions() (trans- late.convert.convert.ArchiveConvertOptionParser
getoutputname() (trans-	method), 234 getpassthroughoptions() (trans-
late.filters.pofilter.FilterOptionParser method), 349	late.convert.convert.ConvertOptionParser
getoutputname() (trans- late.misc.optrecurse.RecursiveOptionParser	method), 238 getpassthroughoptions() (trans-
method), 391	late.convert.po2moz.MozConvertOptionParser
getoutputname() (trans-	method), 250
late.tools.poconflicts.ConflictOptionParser method), 684	getpassthroughoptions() (trans- late.convert.po2tmx.TmxOptionParser
getoutputname() (trans- late.tools.pogrep.GrepOptionParser method),	<pre>method), 255 getpassthroughoptions() (trans-</pre>
1 0 1 1 1	2 - (

late.convert.po2wordfast.WfOptionParser	method), 471
method), 260	<pre>getprojectstyle() (translate.storage.mo.mofile</pre>
<pre>getpassthroughoptions() (trans-</pre>	method), 477
late.filters.pofilter.FilterOptionParser method),	getprojectstyle() (trans-
349	late.storage.mozilla_lang.LangStore method),
<pre>getpassthroughoptions() (trans-</pre>	483
late.misc.optrecurse.RecursiveOptionParser	getprojectstyle() (trans-
method), 391	late.storage.omegat.OmegaTFile method),
getpassthroughoptions() (trans-	488
late.tools.poconflicts.ConflictOptionParser	getprojectstyle() (trans-
method), 684	late.storage.omegat.OmegaTFileTab method),
getpassthroughoptions() (trans-	490
late.tools.pogrep.GrepOptionParser method),	getprojectstyle() (translate.storage.php.phpfile
688	method), 548
getpassthroughoptions() (trans-	getprojectstyle() (trans-
late. tools. por estructure. Split Option Parser	late.storage.pocommon.pofile method), 553
method), 691	getprojectstyle() (trans-
<pre>getpassthroughoptions() (trans-</pre>	late.storage.poheader.poheader method),
late.tools.poterminology.TerminologyOptionPar.	ser 558
method), 695	getprojectstyle() (trans-
getpos() (translate.storage.html.htmlfile method), 436	late.storage.poxliff.PoXliffFile method), 561
<pre>getpos() (translate.storage.html.POHTMLParser</pre>	getprojectstyle() (trans-
method), 433	late.storage.properties.javafile method),
<pre>getprojectstyle() (trans-</pre>	575
late.storage.base.TranslationStore method),	getprojectstyle() (trans-
408	late.storage.properties.javautf8file method),
getprojectstyle() (trans-	577
late.storage.catkeys.CatkeysFile method),	getprojectstyle() (trans-
415	late.storage.properties.joomlafile method),
getprojectstyle() (trans-	579
late.storage.csvl10n.csvfile method), 420	getprojectstyle() (trans-
getprojectstyle() (translate.storage.dtd.dtdfile	late.storage.properties.propfile method),
method), 427	581
<pre>getprojectstyle() (translate.storage.html.htmlfile</pre>	getprojectstyle() (trans-
method), 436	late.storage.properties.stringsfile method),
<pre>getprojectstyle() (trans-</pre>	585
late.storage.html.POHTMLParser method),	getprojectstyle() (trans-
433	late.storage.properties.stringsutf8file method),
<pre>getprojectstyle() (translate.storage.ical.icalfile</pre>	587
method), 442	<pre>getprojectstyle() (translate.storage.pypo.pofile</pre>
<pre>getprojectstyle() (translate.storage.ini.inifile</pre>	method), 589
method), 446	<pre>getprojectstyle() (translate.storage.qm.qmfile</pre>
getprojectstyle() (trans-	method), 596
late.storage.jsonl10n.118NextFile method),	getprojectstyle() (translate.storage.qph.QphFile
452	method), 601
<pre>getprojectstyle() (trans-</pre>	<pre>getprojectstyle() (translate.storage.rc.rcfile</pre>
late.storage.jsonl10n.JsonFile method), 457	method), 606
<pre>getprojectstyle() (trans-</pre>	getprojectstyle() (trans-
late.storage.jsonl10n.JsonNestedFile method),	late.storage.subtitles.AdvSubStationAlphaFile
459	method), 613
getprojectstyle() (trans-	getprojectstyle() (trans-
late.storage.jsonl10n.WebExtensionJsonFile	late.storage.subtitles.MicroDVDFile method),
method), 467	615
<pre>getprojectstyle() (translate.storage.lisa.LISAfile</pre>	getprojectstyle() (trans-
	· ·

late.storage.subtitles.SubRipFile method), 616	getsourcelanguage() (trans- late.storage.jsonl10n.JsonFile method), 457
getprojectstyle() (trans- late.storage.subtitles.SubStationAlphaFile method), 618	<pre>getsourcelanguage() (trans- late.storage.jsonl10n.JsonNestedFile method), 459</pre>
getprojectstyle() (trans- late.storage.subtitles.SubtitleFile method), 620	getsourcelanguage() (trans- late.storage.jsonl10n.WebExtensionJsonFile method), 467
getprojectstyle() (translate.storage.tbx.tbxfile method), 624	getsourcelanguage() (trans- late.storage.lisa.LISAfile method), 471
getprojectstyle() (translate.storage.tiki.TikiStore method), 630	getsourcelanguage() (translate.storage.mo.mofile method), 477
getprojectstyle() (translate.storage.tmx.tmxfile method), 635	<pre>getsourcelanguage() (trans- late.storage.mozilla_lang.LangStore method),</pre>
getprojectstyle() (trans-	483
late.storage.trados.TradosTxtTmFile method), 644	<pre>getsourcelanguage() (trans- late.storage.omegat.OmegaTFile method),</pre>
getprojectstyle() (translate.storage.ts2.tsfile	488
method), 646 getprojectstyle() (translate.storage.txt.TxtFile	getsourcelanguage() (trans- late.storage.omegat.OmegaTFileTab method),
<pre>method), 652 getprojectstyle() (translate.storage.utx.UtxFile     method), 657</pre>	
getprojectstyle() (trans-	late.storage.php.phpfile method), 548 getsourcelanguage() (trans-
late.storage.wordfast.WordfastTMFile method),	late.storage.pocommon.pofile method), 553
667	getsourcelanguage() (trans-
getprojectstyle() (translate.storage.xliff.xlifffile	$late. storage. pox liff. PoX liff File\ method), 561$
<pre>method), 673 getrestype() (translate.storage.poxliff.PoXliffUnit     method), 565</pre>	getsourcelanguage() (trans- late.storage.properties.javafile method), 575
getrestype() (translate.storage.xliff.xliffunit	getsourcelanguage() (trans-
<pre>method), 676 getsourcelanguage() (trans-</pre>	late.storage.properties.javautf8file method), 577
<pre>getsourcelanguage() (trans- late.storage.base.TranslationStore method),</pre>	getsourcelanguage() (trans-
409	late.storage.properties.joomlafile method),
getsourcelanguage() (trans-	579
late.storage.catkeys.CatkeysFile method),	getsourcelanguage() (trans-
415 getsourcelanguage() (trans-	late.storage.properties.propfile method), 581
late.storage.csvl10n.csvfile method), 420	getsourcelanguage() (trans-
<pre>getsourcelanguage() (translate.storage.dtd.dtdfile</pre>	late.storage.properties.stringsfile method), 585
getsourcelanguage() (trans- late.storage.html.htmlfile method), 436	<pre>getsourcelanguage() (trans- late.storage.properties.stringsutf8file method),</pre>
getsourcelanguage() (trans-	587
late.storage.html.POHTMLParser method), 433	getsourcelanguage() (trans- late.storage.pypo.pofile method), 589
getsourcelanguage() (trans- late.storage.ical.icalfile method), 442	getsourcelanguage() (translate.storage.qm.qmfile method), 596
getsourcelanguage() (translate.storage.ini.inifile	getsourcelanguage() (trans-
method), 447 getsourcelanguage() (trans-	late.storage.qph.QphFile method), 601 getsourcelanguage() (translate.storage.rc.rcfile
late.storage.jsonl10n.I18NextFile method), 452	<pre>method), 606 getsourcelanguage() (trans-</pre>

late.storage.subtitles.AdvSubStationAlphaFile method), 613	<pre>gettarget() (translate.storage.tbx.tbxunit method), 627</pre>
<pre>getsourcelanguage() (trans- late.storage.subtitles.MicroDVDFile method),</pre>	<pre>gettarget() (translate.storage.tmx.tmxunit method), 637</pre>
615	<pre>gettarget() (translate.storage.ts2.tsunit method),</pre>
<pre>getsourcelanguage() (trans-</pre>	649
late.storage.subtitles.SubRipFile method), 616	<pre>gettarget() (translate.storage.xliff.xliffunit method), 676</pre>
<pre>getsourcelanguage() (trans-</pre>	gettargetlanguage() (trans-
late.storage.subtitles.SubStationAlphaFile method), 618	late.storage.base.TranslationStore method), 409
<pre>getsourcelanguage() (trans-</pre>	gettargetlanguage() (trans-
late.storage.subtitles.SubtitleFile method), 620	late.storage.catkeys.CatkeysFile method), 415
getsourcelanguage() (translate.storage.tbx.tbxfile	gettargetlanguage() (trans-
method), 625	late.storage.csvl10n.csvfile method), 421
getsourcelanguage() (trans- late.storage.tiki.TikiStore method), 630	gettargetlanguage() (translate.storage.dtd.dtdfile method), 427
<pre>getsourcelanguage() (trans- late.storage.tmx.tmxfile method), 635</pre>	gettargetlanguage() (trans- late.storage.html.htmlfile method), 436
getsourcelanguage() (trans-	gettargetlanguage() (trans-
late.storage.trados.TradosTxtTmFile method), 644	late.storage.html.POHTMLParser method), 433
<pre>getsourcelanguage() (translate.storage.ts2.tsfile</pre>	gettargetlanguage() (trans-
method), 646	late.storage.ical.icalfile method), 442
<pre>getsourcelanguage() (trans- late.storage.txt.TxtFile method), 652</pre>	<pre>gettargetlanguage() (translate.storage.ini.inifile</pre>
getsourcelanguage() (trans-	gettargetlanguage() (trans-
late.storage.utx.UtxFile method), 657	late.storage.jsonl10n.I18NextFile method),
getsourcelanguage() (trans-	452
late.storage.wordfast.WordfastTMFile method), 667	gettargetlanguage() (trans- late.storage.jsonl10n.JsonFile method), 457
getsourcelanguage() (trans-	gettargetlanguage() (trans-
late.storage.xliff.xlifffile method), 673	late.storage.jsonl10n.JsonNestedFile method),
getstartlength() (translate.search.match.matcher	459
method), 406	gettargetlanguage() (trans-
getstartlength() (trans-	late.storage.jsonl10n.WebExtensionJsonFile
late.search.match.terminologymatcher method), 406	method), 467 gettargetlanguage() (trans-
getstoplength() (translate.search.match.matcher	late.storage.lisa.LISAfile method), 471
method), 406	gettargetlanguage() (translate.storage.mo.mofile
getstoplength() (trans-	method), 477
late.search.match.terminologymatcher	gettargetlanguage() (trans-
method), 406	late.storage.mozilla_lang.LangStore method),
${\tt getsubfilename()} \ \ \textit{(translate.storage.oo.oomultifile}$	483
method), 495	gettargetlanguage() (trans-
<pre>getsubfilesrc() (translate.storage.oo.oomultifile</pre>	late.storage.omegat.OmegaTFile method), 488
gettarget() (translate.storage.lisa.LISAunit	gettargetlanguage() (trans-
method), 474	late.storage.omegat.OmegaTFileTab method),
gettarget() (translate.storage.poxliff.PoXliffUnit	490
method), 565	gettargetlanguage() (trans-
gettarget() (translate.storage.qph.QphUnit method), 603	<pre>late.storage.php.phpfile method), 548 gettargetlanguage() (trans-</pre>
тетои), 003	geocargeeranguage() (nans-

late.storage.pocommon.pofile method	), 553	late.storage.trados.TradosTxtTmFile  method),
<pre>gettargetlanguage()</pre>	(trans-	644
late.storage.poheader.poheader 558	method),	<pre>gettargetlanguage() (translate.storage.ts2.tsfile     method), 646</pre>
<pre>gettargetlanguage()</pre>	(trans-	gettargetlanguage() (trans-
late.storage.poxliff.PoXliffFile method	d), 561	late.storage.txt.TxtFile method), 652
<pre>gettargetlanguage()</pre>	(trans-	<pre>gettargetlanguage() (trans-</pre>
late.storage.properties.javafile	method),	late.storage.utx.UtxFile method), 657
576		<pre>gettargetlanguage() (trans-</pre>
<pre>gettargetlanguage()</pre>	(trans-	late.storage.wordfast.WordfastTMFile method),
late.storage.properties.javautf8file	method),	667
577	, ,	gettargetlanguage() (trans-
<pre>gettargetlanguage()</pre>	(trans-	late.storage.xliff.xlifffile method), 674
late.storage.properties.joomlafile	method),	gettargetlen() (trans-
579	memou),	late.storage.base.TranslationUnit method),
gettargetlanguage()	(trans-	411
late.storage.properties.propfile	method),	
581	memoa),	
	(4	late.storage.catkeys.CatkeysUnit method),
<pre>gettargetlanguage()</pre>	(trans-	418
late.storage.properties.stringsfile 585	method),	gettargetlen() (translate.storage.csvl10n.csvunit method), 423
<pre>gettargetlanguage()</pre>	(trans-	<pre>gettargetlen() (translate.storage.dtd.dtdunit</pre>
late.storage.properties.stringsutf8file	method),	method), 429
587		<pre>gettargetlen() (translate.storage.html.htmlunit</pre>
<pre>gettargetlanguage()</pre>	(trans-	method), 438
late.storage.pypo.pofile method), 589		<pre>gettargetlen() (translate.storage.ical.icalunit</pre>
gettargetlanguage() (translate.storage.		method), 444
method), 596	1 1 3	gettargetlen() (translate.storage.ini.iniunit
<pre>gettargetlanguage()</pre>	(trans-	method), 449
late.storage.qph.QphFile method), 60	`	gettargetlen() (trans-
gettargetlanguage() (translate.storage method), 606		late.storage.jsonl10n.118NextUnit method),
gettargetlanguage()	(trans-	gettargetlen() (trans-
late.storage.subtitles.AdvSubStationA	,	late.storage.jsonl10n.JsonNestedUnit method),
	ірнагие	461
method), 613	(tuana	
gettargetlanguage()	(trans-	gettargetlen() (trans-
late.storage.subtitles.MicroDVDFile	тетоа),	late.storage.jsonl10n.JsonUnit method),
615	,	464
gettargetlanguage()	(trans-	gettargetlen() (trans-
late.storage.subtitles.SubRipFile 616	method),	late.storage.jsonl10n.WebExtensionJsonUnit method), 469
<pre>gettargetlanguage()</pre>	(trans-	<pre>gettargetlen() (translate.storage.lisa.LISAunit</pre>
late.storage.subtitles.SubStationAlpha	ıFile	method), 474
method), 618		<pre>gettargetlen() (translate.storage.mo.mounit</pre>
<pre>gettargetlanguage()</pre>	(trans-	method), 480
late.storage.subtitles.SubtitleFile	method),	gettargetlen() (trans-
620	,,	late.storage.mozilla_lang.LangUnit method),
gettargetlanguage() (translate.storage.	thy thyfile	485
method), 625	ιολ.ιολμιε	
**	(trans	gettargetlen() (trans- late.storage.omegat.OmegaTUnit method),
gettargetlanguage()	(trans-	
late.storage.tiki.TikiStore method), 63	,	492 (translate storage pln plnywit
gettargetlanguage()	(trans-	gettargetlen() (translate.storage.php.phpunit
late.storage.tmx.tmxfile method), 635	( -	method), 550
gettargetlanguage()	(trans-	gettargetlen() (trans-

late.storage.pocommon.pounit method), 556	gettemplatename() (trans- late.filters.pofilter.FilterOptionParser method),
gettargetlen() (trans-	349
late.storage.poxliff.PoXliffUnit method),	gettemplatename() (trans-
565	late.misc.optrecurse.RecursiveOptionParser
gettargetlen() (trans-	method), 391
late.storage.properties.propunit method),	gettemplatename() (trans-
583	late.tools.poconflicts.ConflictOptionParser
gettargetlen() (translate.storage.pypo.pounit	method), 684 gettemplatename() (trans-
method), 592	5 I · · · · · · · · · · ·
gettargetlen() (translate.storage.qm.qmunit	late.tools.pogrep.GrepOptionParser method),
method), 598	688
gettargetlen() (translate.storage.qph.QphUnit	gettemplatename() (trans-
method), 603	late.tools.porestructure.SplitOptionParser
gettargetlen() (translate.storage.rc.rcunit	method), 691
method), $609$	gettemplatename() (trans-
gettargetlen() (trans-	late.tools.poterminology.TerminologyOptionParser
$late. storage. subtitles. Subtitle Unit \\ method),$	method), 695
622	<pre>getText() (in module translate.misc.xml_helpers),</pre>
<pre>gettargetlen() (translate.storage.tbx.tbxunit</pre>	396
method), 627	gettext() (translate.storage.oo.ooline method), 495
<pre>gettargetlen() (translate.storage.tiki.TikiUnit</pre>	<pre>gettext_country() (in module translate.lang.data),</pre>
method), 632	360
<pre>gettargetlen() (translate.storage.tmx.tmxunit</pre>	<pre>gettext_domain() (in module translate.lang.data),</pre>
method), 637	360
gettargetlen() (trans-	<pre>gettext_lang() (in module translate.lang.data), 360</pre>
late.storage.trados.TradosUnit method),	gettranslatorcomments() (trans-
641	· · · · · · · · · · · · · · · · ·
	late.storage.poxliff.PoXliffUnit method),
gettargetlen() (translate.storage.ts2.tsunit	565
method), 649	getunits() (translate.storage.base.TranslationStore
gettargetlen() (translate.storage.txt.TxtUnit	method), 409
method), 654	getunits() (translate.storage.base.TranslationUnit
gettargetlen() (translate.storage.utx.UtxUnit	method), 411
method), 659	getunits() (translate.storage.catkeys.CatkeysFile
gettargetlen() (trans-	method), 415
late.storage.wordfast.WordfastUnit method),	getunits() (translate.storage.catkeys.CatkeysUnit
670	method), 418
<pre>gettargetlen() (translate.storage.xliff.xliffunit</pre>	getunits() (translate.storage.csvl10n.csvfile
method), 676	method), 421
gettemplatename() (trans-	getunits() (translate.storage.csvl10n.csvunit
late.convert.convert.ArchiveConvertOptionParse	-
method), 235	getunits() (translate.storage.directory.Directory
gettemplatename() (trans-	method), 425
late.convert.convert.ConvertOptionParser	getunits() (translate.storage.dtd.dtdfile method), 427
method), 238	getunits() (translate.storage.dtd.dtdunit method),
gettemplatename() (trans-	429
late.convert.po2moz.MozConvertOptionParser	getunits() (translate.storage.html.htmlfile method),
	436
method), 250	
gettemplatename() (trans-	getunits() (translate.storage.html.htmlunit method),
late.convert.po2tmx.TmxOptionParser	439
method), 255	getunits() (translate.storage.html.POHTMLParser
gettemplatename() (trans-	method), 433
late.convert.po2wordfast.WfOptionParser	getunits() (translate.storage.ical.icalfile method),
method), 260	442

getunits() (translate.storage.ical.icalunit method),	method), 579
444 getunits() (translate.storage.ini.inifile method), 447	getunits() (translate.storage.properties.propfile method), 581
getunits () (translate.storage.ini.iniunit method), 449	getunits() (translate.storage.properties.propunit
getunits() (translate.storage.jsonl10n.118NextFile	method), 583
method), 452	getunits() (translate.storage.properties.stringsfile
getunits() (translate.storage.jsonl10n.118NextUnit	method), 585
method), 455	getunits()(translate.storage.properties.stringsutf8file
getunits() (translate.storage.jsonl10n.JsonFile	method), 587
method), 457	getunits() (translate.storage.pypo.pofile method),
getunits()(translate.storage.jsonl10n.JsonNestedFile	589
method), 459	getunits() (translate.storage.pypo.pounit method),
getunits() (translate.storage.jsonl10n.JsonNestedUnit method), 461	getunits() (translate.storage.qm.qmfile method), 596
getunits() (translate.storage.jsonl10n.JsonUnit	
method), 464	598
getunits()(translate.storage.jsonl10n.WebExtensionJs	
method), 467	601
getunits()(translate.storage.jsonl10n.WebExtensionJs	
method), 469	604
getunits() (translate.storage.lisa.LISAfile method), 472	getunits() (translate.storage.rc.rcfile method), 606
getunits() (translate.storage.lisa.LISAunit method),	getunits() (translate.storage.rc.rcunit method), 609 getunits() (translate.storage.statistics.Statistics
474	method), 611
getunits() (translate.storage.mo.mofile method), 477	getunits() (translate.storage.subtitles.AdvSubStationAlphaFile
getunits() (translate.storage.mo.mounit method),	method), 613
480	getunits()(translate.storage.subtitles.MicroDVDFile
<pre>getunits()(translate.storage.mozilla_lang.LangStore</pre>	method), 615
method), 483	getunits() (translate.storage.subtitles.SubRipFile
getunits() (translate.storage.mozilla_lang.LangUnit	method), 616
method), 485	getunits() (translate.storage.subtitles.SubStationAlphaFile
getunits() (translate.storage.omegat.OmegaTFile method), 488	method), 618 getunits() (translate.storage.subtitles.SubtitleFile
getunits()(translate.storage.omegat.OmegaTFileTab	method), 620
method), 490	getunits() (translate.storage.subtitles.SubtitleUnit
getunits() (translate.storage.omegat.OmegaTUnit	method), 622
method), 492	getunits() (translate.storage.tbx.tbxfile method), 625
<pre>getunits() (translate.storage.php.phpfile method),</pre>	getunits() (translate.storage.tbx.tbxunit method),
548	627
getunits() (translate.storage.php.phpunit method),	getunits() (translate.storage.tiki.TikiStore method),
550 (translate atomace recommen refle	630
getunits() (translate.storage.pocommon.pofile method), 553	getunits() (translate.storage.tiki.TikiUnit method), 632
getunits() (translate.storage.pocommon.pounit	getunits() (translate.storage.tmx.tmxfile method),
method), 556	635
getunits() (translate.storage.poxliff.PoXliffFile	getunits() (translate.storage.tmx.tmxunit method),
method), 561	638
<pre>getunits() (translate.storage.poxliff.PoXliffUnit</pre>	<pre>getunits() (translate.storage.trados.TradosTxtTmFile</pre>
method), 565	method), 644
getunits() (translate.storage.properties.javafile	getunits() (translate.storage.trados.TradosUnit
method), 576	method), 642
getunits() (translate.storage.properties.javautf8file method), 577	getunits() (translate.storage.ts2.tsfile method), 646 getunits() (translate.storage.ts2.tsunit method), 649
getunits() (translate.storage.properties.joomlafile	getunits() (translate.storage.tsz.tsunu memoa), 049 getunits() (translate.storage.txt.TxtFile method),
500 and the state of the state	gooding of the memory,

652	getusagestring() (trans-
getunits() (translate.storage.txt.TxtUnit method), 654	late.convert.convert.ConvertOptionParser method), 238
<pre>getunits() (translate.storage.utx.UtxFile method),</pre>	
657	late.convert.po2moz.MozConvertOptionParser
getunits() (translate.storage.utx.UtxUnit method),	method), 250
659	getusagestring() (trans-
<pre>getunits() (translate.storage.wordfast.WordfastTMFil</pre>	e late.convert.po2tmx.TmxOptionParser method), 255
getunits() (translate.storage.wordfast.WordfastUnit	getusagestring() (trans-
method), 670	late.convert.po2wordfast.WfOptionParser
getunits() (translate.storage.xliff.xlifffile method),	method), 260
674	getusagestring() (trans-
getunits() (translate.storage.xliff.xliffunit method), 677	late.filters.pofilter.FilterOptionParser method), 349
<pre>getunits() (translate.storage.zip.ZIPFile method),</pre>	getusagestring() (trans-
681	late.misc.optrecurse.RecursiveOptionParser
geturls() (in module translate.filters.decoration), 347	method), 391
getusageman() (trans-	getusagestring() (trans-
late.convert.convert.ArchiveConvertOptionParse method), 235	r late.tools.poconflicts.ConflictOptionParser method), 684
getusageman() (trans-	getusagestring() (trans-
late.convert.convert.ConvertOptionParser	late.tools.pogrep.GrepOptionParser method),
method), 238	688
getusageman() (trans-	getusagestring() (trans-
late.convert.po2moz.MozConvertOptionParser method), 250	late.tools.porestructure.SplitOptionParser method), 691
getusageman() (trans-	getusagestring() (trans-
late.convert.po2tmx.TmxOptionParser	late.tools.poterminology.TerminologyOptionParser
method), 255	method), 695
getusageman() (trans-	<pre>getvalue() (translate.storage.jsonl10n.I18NextUnit</pre>
late.convert.po2wordfast.WfOptionParser	method), 455
method), 260	<pre>getvalue() (translate.storage.jsonl10n.JsonNestedUnit</pre>
getusageman() (trans-	method), 461
late.filters.pofilter.FilterOptionParser method), 349	getvalue() (translate.storage.jsonl10n.JsonUnit method), 464
getusageman() (trans-	<pre>getvalue() (translate.storage.jsonl10n.WebExtensionJsonUnit</pre>
late.misc.opt recurse. Recursive Option Parser	method), 469
method), 391	getvariables() (in module trans-
getusageman() (trans-	late.filters.decoration), 347
late. tools. poconflicts. Conflict Option Parser	getXMLlang() (in module trans-
method), 684	late.misc.xml_helpers), 396
getusageman() (trans-	getXMLspace() (in module trans-
late.tools.pogrep.GrepOptionParser method),	late.misc.xml_helpers), 396
688	git (class in translate.storage.versioncontrol.git), 664
getusageman() (trans-	GnomeChecker (class in translate.filters.checks), 281
late.tools.porestructure.SplitOptionParser	GrepMatch (class in translate.tools.pogrep), 687
method), 691	GrepOptionParser (class in translate.tools.pogrep),
getusageman() (trans-	687
late.tools.poterminology.TerminologyOptionPars	
method), 695	guess_encoding() (translate.storage.html.htmlfile
getusagestring() (trans-	method), 436
late.convert.convert.ArchiveConvertOptionParse	
method), 235	late.storage.html.POHTMLParser method),

433 guess_language() (in module translate.lang.team),	hasplural() (translate.storage.jsonl10n.JsonUnit method), 464
381	hasplural() (trans-
Н	late.storage.jsonl10n.WebExtensionJsonUnit method), 469
handle_charref() (translate.storage.html.htmlfile method), 436	hasplural() (translate.storage.lisa.LISAunit method), 474
handle_charref() (trans- late.storage.html.POHTMLParser method),	hasplural() (translate.storage.mo.mounit method), 480
433	hasplural() (trans-
handle_entityref() (trans- late.storage.html.htmlfile method), 436	late.storage.mozilla_lang.LangUnit method), 485
handle_entityref() (trans- late.storage.html.POHTMLParser method), 433	hasplural() (translate.storage.omegat.OmegaTUnit method), 492 hasplural() (translate.storage.php.phpunit method),
handlecsvunit() (translate.convert.csv2po.csv2po method), 241	550 hasplural() (translate.storage.pocommon.pounit
has_content (trans-	method), 556
late.storage.placeables.strelem.StringElem attribute), 527	hasplural() (translate.storage.poxliff.PoXliffUnit method), 565
has_key() (translate.misc.dictutils.cidict method), 386 has_key() (translate.storage.oo.unormalizechar	hasplural() (translate.storage.properties.propunit method), 583
method), 496	hasplural() (translate.storage.pypo.pounit method),
has_translatable_content() (trans- late.storage.html.htmlfile method), 436	592 hasplural() (translate.storage.qm.qmunit method),
has_translatable_content() (trans-	598
late.storage.html.POHTMLParser method),	hasplural() (translate.storage.qph.QphUnit method), 604
has_translatable_text (trans-	hasplural() (translate.storage.rc.rcunit method), 609
late.storage.xml_extract.extract.Translatable attribute), 678	hasplural() (translate.storage.subtitles.SubtitleUnit method), 622
HashProgressBar (class in trans- late.misc.progressbar), 394	hasplural() (translate.storage.tbx.tbxunit method), 627
	hasplural() (translate.storage.tiki.TikiUnit method), 632
hasplural() (translate.storage.base.TranslationUnit method), 411	hasplural() (translate.storage.tmx.tmxunit method), 638
	hasplural() (translate.storage.trados.TradosUnit method), 642
hasplural() (translate.storage.csvl10n.csvunit method), 423	hasplural() (translate.storage.ts2.tsunit method), 649
hasplural() (translate.storage.dtd.dtdunit method), 429	hasplural() (translate.storage.txt.TxtUnit method), 654
hasplural() (translate.storage.html.htmlunit method), 439	hasplural() (translate.storage.utx.UtxUnit method), 660
hasplural() (translate.storage.ical.icalunit method), 444	hasplural() (trans- late.storage.wordfast.WordfastUnit method),
hasplural() (translate.storage.ini.iniunit method), 449	670 hasplural() (translate.storage.xliff.xliffunit method),
hasplural() (translate.storage.jsonl10n.118NextUnit	677
method), 455	hassuggestion() (trans-
hasplural() (trans-	late.filters.checks.StandardUnitChecker method), 338
late.storage.jsonl10n.JsonNestedUnit method), 461	hastypecomment() (translate.storage.pypo.pounit

method), 592	411
he (class in translate.lang.he), 368	infer_state() (trans-
header (translate.storage.wordfast.WordfastHeader at	<del>_</del> , , , , , , , , , , , , , , , , , , ,
tribute), 666	418
header() (translate.storage.mo.mofile method), 477	<pre>infer_state() (translate.storage.csvl10n.csvunit</pre>
header() (translate.storage.pocommon.pofile method)	_ , ,
553	<pre>infer_state() (translate.storage.dtd.dtdunit</pre>
header() (translate.storage.poheader.poheade	
method), 558	<pre>infer_state() (translate.storage.html.htmlunit</pre>
header() (translate.storage.poxliff.PoXliffFile	
method), 561	<pre>infer_state() (translate.storage.ical.icalunit</pre>
header() (translate.storage.pypo.pofile method), 589	method), 444
hg (class in translate.storage.versioncontrol.hg), 664	<pre>infer_state() (translate.storage.ini.iniunit method),</pre>
hi (class in translate.lang.hi), 369	449
htmlentitydecode() (in module trans	
late.misc.quote), 395	late.storage.jsonl10n.118NextUnit method),
<del>-</del>	
<u>-</u>	
late.misc.quote), 395	infer_state() (trans-
htmlfile (class in translate.storage.html), 434	late.storage.jsonl10n.JsonNestedUnit method),
htmlunit (class in translate.storage.html), 437	461
hy (class in translate.lang.hy), 370	<pre>infer_state() (translate.storage.jsonl10n.JsonUnit</pre>
I	method), 464
1	infer_state() (trans-
I18NextFile (class in translate.storage.jsonl10n), 451	
I18NextUnit (class in translate.storage.jsonl10n), 453	
ical2po (class in translate.convert.ical2po), 242	<pre>infer_state() (translate.storage.lisa.LISAunit</pre>
icalfile (class in translate.storage.ical), 441	method), 474
icalunit (class in translate.storage.ical), 443	<pre>infer_state() (translate.storage.mo.mounit</pre>
ignoretests (translate.lang.common.Common	n method), 480
attribute), 358	infer_state() (trans-
inc2po() (in module trans	late.storage.mozilla_lang.LangUnit method),
late.convert.mozfunny2prop), 244	485
inc2prop() (in module trans	_ infer_state() (trans-
late.convert.mozfunny2prop), 244	late.storage.omegat.OmegaTUnit method),
INCLUDEATTRS (translate.storage.html.htmlfile at	402
tribute), 435	<pre>infer_state() (translate.storage.php.phpunit</pre>
index() (translate.misc.multistring.multistring	1 0 550
Tildez () (ranstate:mise:mutistring:mutistring	
method) 388	
method), 388	infer_state() (translate.storage.pocommon.pounit
INDEX DIRECTORY NAME (trans	infer_state() (translate.storage.pocommon.pounit method), 556
INDEX_DIRECTORY_NAME (trans late.search.indexing.CommonIndexer.Common	infer_state() (translate.storage.pocommon.pounit method), 556 Databa@r_state() (translate.storage.poxliff.PoXliffUnit
INDEX_DIRECTORY_NAME (trans late.search.indexing.CommonIndexer.Common attribute), 397	infer_state() (translate.storage.pocommon.pounit method), 556 Databaser_state() (translate.storage.poxliff.PoXliffUnit method), 565
<pre>INDEX_DIRECTORY_NAME</pre>	infer_state() (translate.storage.pocommon.pounit method), 556 Databaser_state() (translate.storage.poxliff.PoXliffUnit method), 565 infer_state() (trans-
INDEX_DIRECTORY_NAME (translate.search.indexing.CommonIndexer.Common.attribute), 397  index_document() (translate.search.indexing.CommonIndexer.CommonIndexe	infer_state() (translate.storage.pocommon.pounit method), 556 Databaser_state() (translate.storage.poxliff.PoXliffUnit method), 565 infer_state() (trans- Database late.storage.properties.propunit method),
INDEX_DIRECTORY_NAME (trans late.search.indexing.CommonIndexer.Common.attribute), 397  index_document() (trans late.search.indexing.CommonIndexer.Common.method), 398	infer_state() (translate.storage.pocommon.pounit method), 556 Databaser_state() (translate.storage.poxliff.PoXliffUnit method), 565 infer_state() (trans- Database late.storage.properties.propunit method), 583
INDEX_DIRECTORY_NAME (trans late.search.indexing.CommonIndexer.Common attribute), 397  index_document() (trans late.search.indexing.CommonIndexer.Common method), 398  index_document() (trans late.search.indexing.CommonIndexer.Common method), 398	infer_state() (translate.storage.pocommon.pounit method), 556 Databaser_state() (translate.storage.poxliff.PoXliffUnit method), 565 infer_state() (translate.storage.propunit method), 583 infer_state() (translate.storage.pypo.pounit
INDEX_DIRECTORY_NAME (trans late.search.indexing.CommonIndexer.Common attribute), 397  index_document() (trans late.search.indexing.CommonIndexer.Common method), 398  index_document() (trans late.search.indexing.PyLuceneIndexer.PyLuceneIn	infer_state() (translate.storage.pocommon.pounit method), 556 Databaser_state() (translate.storage.poxliff.PoXliffUnit method), 565 infer_state() (translate.storage.propunit method), 583 infer_state() (translate.storage.pypo.pounit meDatabase method), 592
INDEX_DIRECTORY_NAME (trans late.search.indexing.CommonIndexer.Common attribute), 397  index_document() (trans late.search.indexing.CommonIndexer.Common method), 398  index_document() (trans late.search.indexing.PyLuceneIndexer.PyLucene method), 401	infer_state() (translate.storage.pocommon.pounit  method), 556  Databaser_state() (translate.storage.poxliff.PoXliffUnit method), 565  infer_state() (translate.storage.propunit method), 583  infer_state() (translate.storage.pypo.pounit meDatabase method), 592 infer_state() (translate.storage.qm.qmunit
INDEX_DIRECTORY_NAME (trans late.search.indexing.CommonIndexer.Common attribute), 397  index_document() (trans late.search.indexing.CommonIndexer.Common method), 398  index_document() (trans late.search.indexing.PyLuceneIndexer.PyLucene method), 401  index_document() (trans late.search.indexing.PyLuceneIndexer.PyLucene method), 401	infer_state() (translate.storage.pocommon.pounit  method), 556  Databaser_state() (translate.storage.poxliff.PoXliffUnit method), 565  infer_state() (translate.storage.propunit method), 583  infer_state() (translate.storage.pypo.pounit meDatabase method), 592 infer_state() (translate.storage.qm.qmunit method), 598
INDEX_DIRECTORY_NAME  late.search.indexing.CommonIndexer.Common. attribute), 397  index_document() (trans late.search.indexing.CommonIndexer.Common. method), 398  index_document() (trans late.search.indexing.PyLuceneIndexer.PyLucen. method), 401  index_document() (trans late.search.indexing.XapianIndexer.XapianDate	infer_state() (translate.storage.pocommon.pounit  method), 556  Databaser_state() (translate.storage.poxliff.PoXliffUnit method), 565  infer_state() (translate.storage.propunit method), 583  infer_state() (translate.storage.pypo.pounit meDatabase method), 592 infer_state() (translate.storage.qm.qmunit method), 598  abasefer_state() (translate.storage.qph.QphUnit
INDEX_DIRECTORY_NAME  late.search.indexing.CommonIndexer.Common. attribute), 397  index_document() (trans late.search.indexing.CommonIndexer.Common. method), 398  index_document() (trans late.search.indexing.PyLuceneIndexer.PyLucen. method), 401  index_document() (trans late.search.indexing.XapianIndexer.XapianDat. method), 403	infer_state() (translate.storage.pocommon.pounit  method), 556  Database_state() (translate.storage.poxliff.PoXliffUnit  method), 565  infer_state() (translate.storage.propunit method),  583  infer_state() (translate.storage.pypo.pounit  meDatabase method), 592  infer_state() (translate.storage.qm.qmunit  method), 598  abase fer_state() (translate.storage.qph.QphUnit  method), 604
INDEX_DIRECTORY_NAME (trans late.search.indexing.CommonIndexer.Common attribute), 397  index_document() (trans late.search.indexing.CommonIndexer.Common method), 398  index_document() (trans late.search.indexing.PyLuceneIndexer.PyLucene method), 401  index_document() (trans late.search.indexing.XapianIndexer.XapianData method), 403  indicpunc (translate.lang.common.Common at	infer_state() (translate.storage.pocommon.pounit
INDEX_DIRECTORY_NAME (trans late.search.indexing.CommonIndexer.Common attribute), 397  index_document() (trans late.search.indexing.CommonIndexer.Common method), 398  index_document() (trans late.search.indexing.PyLuceneIndexer.PyLucene method), 401  index_document() (trans late.search.indexing.XapianIndexer.XapianDat method), 403  indicpunc (translate.lang.common.Common at tribute), 358	infer_state() (translate.storage.pocommon.pounit  method), 556  Databaser_state() (translate.storage.poxliff.PoXliffUnit method), 565  infer_state() (translate.storage.propunit method), 583  infer_state() (translate.storage.pypo.pounit meDatabase method), 592 infer_state() (translate.storage.qm.qmunit method), 598  abasefer_state() (translate.storage.qph.QphUnit method), 604  infer_state() (translate.storage.rc.rcunit method), 609
INDEX_DIRECTORY_NAME (trans late.search.indexing.CommonIndexer.Common attribute), 397  index_document() (trans late.search.indexing.CommonIndexer.Common method), 398  index_document() (trans late.search.indexing.PyLuceneIndexer.PyLucene method), 401  index_document() (trans late.search.indexing.XapianIndexer.XapianData method), 403  indicpunc (translate.lang.common.Common at	infer_state() (translate.storage.pocommon.pounit  method), 556  Databaser_state() (translate.storage.poxliff.PoXliffUnit method), 565  infer_state() (translate.storage.propunit method), 583  infer_state() (translate.storage.pypo.pounit meDatabase method), 592 infer_state() (translate.storage.qm.qmunit method), 598  abasefer_state() (translate.storage.qph.QphUnit method), 604  infer_state() (translate.storage.rc.rcunit method), 609  infer_state() (translate.storage.rc.rcunit method), 609

622	insert() (translate.storage.placeables.base.Ept
infer_state() (translate.storage.tbx.tbxunit method), 627	method), 500 insert() (translate.storage.placeables.base.Ex
infer_state() (translate.storage.tiki.TikiUnit	method), 508
method), 632	insert() (translate.storage.placeables.base.G
infer_state() (translate.storage.tmx.tmxunit	method), 505
<pre>method), 638 infer_state() (translate.storage.trados.TradosUnit</pre>	insert() (translate.storage.placeables.base.It method), 503
method), 642	insert() (translate.storage.placeables.base.Ph
<pre>infer_state() (translate.storage.ts2.tsunit method),</pre>	method), 501
649	insert() (translate.storage.placeables.base.Sub
infer_state() (translate.storage.txt.TxtUnit method), 654	<pre>method), 511 insert() (translate.storage.placeables.base.X</pre>
infer_state() (translate.storage.utx.UtxUnit	method), 509
method), 660	insert() (translate.storage.placeables.general.AltAttrPlaceable
infer_state() (trans-	method), 513
late.storage.wordfast.WordfastUnit method),	insert() (translate.storage.placeables.general.XMLEntityPlaceable
670 infer_state() (translate.storage.xliff.xliffunit	method), 514 insert() (translate.storage.placeables.general.XMLTagPlaceable
method), 677	method), 516
ini2po (class in translate.convert.ini2po), 243	<pre>insert() (translate.storage.placeables.interfaces.BasePlaceable</pre>
inifile (class in translate.storage.ini), 446	method), 518
init_headers() (translate.storage.mo.mofile	insert() (translate.storage.placeables.interfaces.InvisiblePlaceable
<pre>method), 477 init_headers() (translate.storage.pocommon.pofile</pre>	method), 520 insert() (translate.storage.placeables.interfaces.MaskingPlaceable
method), 553	method), 521
init_headers() (trans-	$\verb insert()  (translate.storage.placeables.interfaces.Replacement Placeables)                                      $
late.storage.poheader.poheader method),	method), 523
558	insert() (translate.storage.placeables.interfaces.SubflowPlaceable
<pre>init_headers() (translate.storage.poxliff.PoXliffFile</pre>	method), 525 insert() (translate.storage.placeables.strelem.StringElem
init_headers() (translate.storage.pypo.pofile	method), 527
method), 590	$\verb insert()  (translate.storage.placeables.terminology.TerminologyPlaceables.terminolog$
initbody() (translate.storage.lisa.LISAfile method),	method), 529
472	insert() (translate.storage.placeables.xliff.Bpt method), 531
initbody() (translate.storage.poxliff.PoXliffFile method), 561	insert() (translate.storage.placeables.xliff.Bx
initbody() (translate.storage.qph.QphFile method),	method), 536
601	insert() (translate.storage.placeables.xliff.Ept
initbody() (translate.storage.tbx.tbxfile method), 625	method), 533
initbody() (translate.storage.tmx.tmxfile method), 635	insert() (translate.storage.placeables.xliff.Ex method), 538
initbody () (translate.storage.ts2.tsfile method), 646	insert() (translate.storage.placeables.xliff.G
initbody() (translate.storage.xliff.xlifffile method),	method), 539
674	<pre>insert() (translate.storage.placeables.xliff.It method),</pre>
inittm() (translate.search.match.matcher method),	541
406 inittm() (translate.search.match.terminologymatcher	insert() (translate.storage.placeables.xliff.Ph method), 544
method), 406	insert() (translate.storage.placeables.xliff.Sub
iniunit ( <i>class in translate.storage.ini</i> ), 448	method), 542
insert() (translate.storage.placeables.base.Bpt	<pre>insert() (translate.storage.placeables.xliff.UnknownXML</pre>
method), 498	method), 546
insert() (translate.storage.placeables.base.Bx method), 506	<pre>insert() (translate.storage.placeables.xliff.X method),</pre>
memou), 500	JJ <del>T</del>

insert_between()	(trans-	insert_between() (trans-
late.storage.placeables.base.Bpt 498	method),	late.storage.placeables.terminology.TerminologyPlaceable method), 529
insert_between()	(trans-	insert_between() (trans-
late.storage.placeables.base.Bx 506	method),	late.storage.placeables.xliff.Bpt method), 531
<pre>insert_between()</pre>	(trans-	insert_between() (trans-
late.storage.placeables.base.Ept 500	method),	late.storage.placeables.xliff.Bx method), 536
insert_between()	(trans-	insert_between() (trans-
late.storage.placeables.base.Ex 508	method),	late.storage.placeables.xliff.Ept method), 533
insert_between()	(trans-	insert_between() (trans-
late.storage.placeables.base.G 505	method),	late.storage.placeables.xliff.Ex method), 538
insert_between()	(trans-	insert_between() (trans-
late.storage.placeables.base.It	method),	late.storage.placeables.xliff.G method), 539
503		insert_between() (trans-
insert_between()	(trans-	late.storage.placeables.xliff.It method), 541
late.storage.placeables.base.Ph 501	method),	insert_between() (trans- late.storage.placeables.xliff.Ph method),
insert_between()	(trans-	544 method),
late.storage.placeables.base.Sub	method),	insert_between() (trans-
511	,	late.storage.placeables.xliff.Sub method),
insert_between()	(trans-	542
late.storage.placeables.base.X 509	method),	<pre>insert_between()</pre>
insert_between()	(trans-	method), 546
late.storage.placeables.general.Alt. method), 513	AttrPlaceable	insert_between() (trans- late.storage.placeables.xliff.X method), 534
<pre>insert_between()</pre>	(trans-	<pre>intuplelist() (in module translate.filters.checks),</pre>
late.storage.placeables.general.XM	<i>ILEntityPlace</i>	
method), 515		InvalidBundleError, 414
insert_between()	(trans- U Taa Plaasab	InvalidStateObjectError, 672  leinvertedpunc (translate.lang.common.Common at-
method), 516		tribute), 358
<pre>insert_between()     late.storage.placeables.interfaces.E</pre>	(trans- RasePlaceable	
method), 518	aser iaceabie	IOSChecker (class in translate.filters.checks), 286
insert_between()	(trans-	is_available() (in module trans-
late.storage.placeables.interfaces.I		
method), 520		is_available() (in module trans-
insert_between()	(trans-	late.storage.versioncontrol.bzr), 663
late.storage.placeables.interfaces. Mathematical states and the states are also as a superior of the states are also as	<i>AaskingPlace</i>	
method), 521		late.storage.versioncontrol.cvs), 664
insert_between()	(trans-	is_available() (in module trans-
method), 523		claceable late.storage.versioncontrol.darcs), 664 is_available() (in module trans-
<pre>insert_between()</pre>	(trans-	late.storage.versioncontrol.git), 664
late.storage.placeables.interfaces.S	ивпоwPlacea	
<pre>method), 525 insert_between()</pre>	(trans-	late.storage.versioncontrol.hg), 665 is_available() (in module trans-
late.storage.placeables.strelem.Stri	`	late.storage.versioncontrol.svn), 665
method), 527		is_comment_end() (in module trans-
<i>''</i>		

late.storage.properties), 574	550
<pre>is_comment_one_line() (in module trans- late.storage.properties), 574</pre>	isblank() (translate.storage.pocommon.pounit method), 556
is_comment_start() (in module trans-	isblank() (translate.storage.poxliff.PoXliffUnit
late.storage.properties), 574	method), 565
is_css_entity() (in module trans- late.convert.dtd2po), 241	isblank() (translate.storage.properties.propunit method), 583
is_line_continuation() (in module trans- late.storage.properties), 575	isblank() (translate.storage.pypo.pounit method), 592
isalnum() (translate.misc.multistring.multistring	isblank() (translate.storage.qm.qmunit method), 598
method), 388	isblank() (translate.storage.qph.QphUnit method),
isalpha() (translate.misc.multistring.multistring	604
method), 388	isblank() (translate.storage.rc.rcunit method), 609
isapproved() (translate.storage.poxliff.PoXliffUnit	isblank() (translate.storage.subtitles.SubtitleUnit
method), 565	method), 622
isapproved() (translate.storage.xliff.xliffunit	isblank() (translate.storage.tbx.tbxunit method), 627
method), 677	isblank() (translate.storage.tiki.TikiUnit method),
isarchive() (trans-	633
late.convert.convert.ArchiveConvertOptionParse	
method), 235	638
isarchive() (trans-	isblank() (translate.storage.trados.TradosUnit
late.convert.po2tmx.TmxOptionParser	method), 642
method), 255	isblank() (translate.storage.ts2.tsunit method), 649
isarchive() (trans-	isblank() (translate.storage.txt.TxtUnit method), 654
late.convert.po2wordfast.WfOptionParser	isblank() (translate.storage.utx.UtxUnit method),
method), 260	660
isblank() (translate.storage.base.TranslationUnit method), 411	isblank() (translate.storage.wordfast.WordfastUnit method), 670
isblank() (translate.storage.catkeys.CatkeysUnit method), 418	isblank() (translate.storage.xliff.xliffunit method), 677
isblank() (translate.storage.csvl10n.csvunit method), 423	isdecimal() (translate.misc.multistring.multistring method), 388
isblank() (translate.storage.dtd.dtdunit method), 429	isdigit() (translate.misc.multistring.multistring
isblank() (translate.storage.html.htmlunit method),	method), 388
439	iseditable (translate.storage.placeables.strelem.StringElem
isblank() (translate.storage.ical.icalunit method),	attribute), 527
444	isempty() (translate.storage.base.TranslationStore
isblank() (translate.storage.ini.iniunit method), 449	method), 409
isblank() (translate.storage.jsonl10n.118NextUnit	isempty() (translate.storage.catkeys.CatkeysFile
method), 455	method), 415
isblank() (translate.storage.jsonl10n.JsonNestedUnit	<pre>isempty() (translate.storage.csvl10n.csvfile method),</pre>
method), 461	421
isblank() (translate.storage.jsonl10n.JsonUnit	isempty() (translate.storage.dtd.dtdfile method), 427
method), 464	isempty() (translate.storage.html.htmlfile method),
isblank() (translate.storage.jsonl10n.WebExtensionJso	
method), 469	<pre>isempty() (translate.storage.html.POHTMLParser</pre>
isblank() (translate.storage.lisa.LISAunit method),	method), 433
474	isempty() (translate.storage.ical.icalfile method), 442
isblank() (translate.storage.mo.mounit method), 481	isempty() (translate.storage.ini.inifile method), 447
isblank() (translate.storage.mozilla_lang.LangUnit	isempty() (translate.storage.jsonl10n.118NextFile
method), 485	method), 452
isblank() (translate.storage.omegat.OmegaTUnit	isempty() (translate.storage.jsonl10n.JsonFile
method), 492	method), 457
isblank() (translate.storage.php.phpunit method),	
	5 July 11 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

method), 459	method), 667
<pre>isempty() (translate.storage.jsonl10n.WebExtensionJso</pre>	
method), 467	isexcluded() (trans-
<pre>isempty() (translate.storage.lisa.LISAfile method),</pre>	late.convert.convert.ArchiveConvertOptionParser
472	method), 235
isempty() (translate.storage.mo.mofile method), 478	isexcluded() (trans-
<pre>isempty() (translate.storage.mozilla_lang.LangStore</pre>	late.convert.convert.ConvertOptionParser
method), 483	method), 238
isempty() (translate.storage.omegat.OmegaTFile	isexcluded() (trans-
method), 488	late.convert.po2moz.MozConvertOptionParser
isempty() (translate.storage.omegat.OmegaTFileTab	method), 250
method), 490	
isempty() (translate.storage.php.phpfile method), 548	late.convert.po2tmx.TmxOptionParser
isempty() (translate.storage.pocommon.pofile	method), 255
method), 553	isexcluded() (trans-
isempty() (translate.storage.poxliff.PoXliffFile	late.convert.po2wordfast.WfOptionParser
method), 561	method), 260
	isexcluded() (trans-
method), 576	$late.filters.pofilter.FilterOptionParser\ method),$
<pre>isempty() (translate.storage.properties.javautf8file</pre>	349
method), 577	isexcluded() (trans-
<pre>isempty() (translate.storage.properties.joomlafile</pre>	late.misc.optrecurse.RecursiveOptionParser
method), 579	method), 391
<pre>isempty() (translate.storage.properties.propfile</pre>	isexcluded() (trans-
method), 581	late.tools.poconflicts.ConflictOptionParser
<pre>isempty() (translate.storage.properties.stringsfile</pre>	method), 684
method), 585	isexcluded() (trans-
<pre>isempty() (translate.storage.properties.stringsutf8file</pre>	late.tools.pogrep.GrepOptionParser method),
method), 587	688
isempty() (translate.storage.pypo.pofile method), 590	
isempty() (translate.storage.qm.qmfile method), 596	late.tools.porestructure.SplitOptionParser
isempty() (translate.storage.qph.QphFile method), 601	method), 691
	isexcluded() (trans-
isempty() (translate.storage.rc.rcfile method), 606	late.tools.poterminology.TerminologyOptionParser
isempty()(translate.storage.subtitles.AdvSubStationAlp	
method), 613	isexcluded() (translate.tools.pydiff.DirDiffer
isempty() (translate.storage.subtitles.MicroDVDFile	method), 698
method), 615	$\verb isfragile  (translate.storage.placeables.strelem.StringElem $
<pre>isempty() (translate.storage.subtitles.SubRipFile</pre>	attribute), 527
method), 616	isfuzzy() (translate.filters.checks.StandardUnitChecker
$\verb isempty()  (translate.storage.subtitles.SubStationAlpha Parameters)    (translate.subtitles.SubStationAlpha Parameters)    (translate.$	File method), 338
method), 618	isfuzzy() (translate.storage.base.TranslationUnit
<pre>isempty() (translate.storage.subtitles.SubtitleFile</pre>	method), 411
method), 620	isfuzzy() (translate.storage.catkeys.CatkeysUnit
<pre>isempty() (translate.storage.tbx.tbxfile method), 625</pre>	method), 418
<pre>isempty() (translate.storage.tiki.TikiStore method),</pre>	isfuzzy() (translate.storage.csvl10n.csvunit method),
630	423
<pre>isempty() (translate.storage.tmx.tmxfile method), 635</pre>	isfuzzy() (translate.storage.dtd.dtdunit method), 429
<pre>isempty() (translate.storage.trados.TradosTxtTmFile</pre>	isfuzzy() (translate.storage.html.htmlunit method),
method), 644	439
isempty() (translate.storage.ts2.tsfile method), 646	isfuzzy() (translate.storage.ical.icalunit method),
isempty() (translate.storage.txt.TxtFile method), 652	444 (transiate.storage.icai.icaianti memoa),
isempty() (translate.storage.txt.TxtFite method), 657	isfuzzy() (translate.storage.ini.iniunit method), 449
isempty() (translate.storage.wordfast.WordfastTMFile	isfuzzy() (translate.storage.jsonl10n.I18NextUnit
Locinpey () (iransiane.siorage.woragasi.woragasi1MTile	LOLUZZY () (HUBBURE SIOTUZE JSOIII TOIL I TOIL EN OHH

method), 455	439
isfuzzy() (translate.storage.jsonl10n.JsonNestedUnit method), 461	isheader() (translate.storage.ical.icalunit method), 444
	isheader() (translate.storage.ini.iniunit method), 449 isheader() (translate.storage.jsonl10n.118NextUnit
isfuzzy() (translate.storage.jsonl10n.WebExtensionJsomethod), 469	
isfuzzy() (translate.storage.lisa.LISAunit method),	method), 461
474 isfuzzy() (translate.storage.mo.mounit method), 481	isheader() (translate.storage.jsonl10n.JsonUnit method), 464
isfuzzy() (translate.storage.mozilla_lang.LangUnit method), 486	isheader() (translate.storage.jsonl10n.WebExtensionJsonUnit method), 469
isfuzzy() (translate.storage.omegat.OmegaTUnit method), 493	isheader() (translate.storage.lisa.LISAunit method), 475
isfuzzy() (translate.storage.php.phpunit method), 550	isheader() (translate.storage.mo.mounit method), 481
isfuzzy() (translate.storage.pocommon.pounit method), 556	isheader() (translate.storage.mozilla_lang.LangUnit method), 486
isfuzzy() (translate.storage.poxliff.PoXliffUnit method), 565	isheader() (translate.storage.omegat.OmegaTUnit method), 493
isfuzzy() (translate.storage.properties.propunit method), 583	isheader() (translate.storage.php.phpunit method), 550
isfuzzy() (translate.storage.pypo.pounit method), 593	isheader() (translate.storage.pocommon.pounit method), 556
isfuzzy() (translate.storage.qm.qmunit method), 598	isheader() (translate.storage.poxliff.PoXliffUnit
isfuzzy() (translate.storage.qph.QphUnit method), 604	<pre>method), 565 isheader() (translate.storage.properties.propunit</pre>
isfuzzy() (translate.storage.rc.rcunit method), 609	method), 583
isfuzzy() (translate.storage.subtitles.SubtitleUnit method), 622	isheader() (translate.storage.pypo.pounit method), 593
isfuzzy() (translate.storage.tbx.tbxunit method), 627 isfuzzy() (translate.storage.tiki.TikiUnit method),	isheader() (translate.storage.qm.qmunit method), 598
633 isfuzzy() (translate.storage.tmx.tmxunit method),	isheader() (translate.storage.qph.QphUnit method), 604
638	
	isheader () (translate.storage.rc.rcunit method), 609
isfuzzy() (translate.storage.trados.TradosUnit method), 642	isheader() (translate.storage.subtitles.SubtitleUnit
method), 642 isfuzzy() (translate.storage.ts2.tsunit method), 649	<pre>isheader() (translate.storage.subtitles.SubtitleUnit</pre>
method), 642 isfuzzy() (translate.storage.ts2.tsunit method), 649 isfuzzy() (translate.storage.txt.TxtUnit method), 654 isfuzzy() (translate.storage.utx.UtxUnit method),	<pre>isheader() (translate.storage.subtitles.SubtitleUnit</pre>
method), 642 isfuzzy() (translate.storage.ts2.tsunit method), 649 isfuzzy() (translate.storage.txt.TxtUnit method), 654 isfuzzy() (translate.storage.utx.UtxUnit method), 660 isfuzzy() (translate.storage.wordfast.WordfastUnit	<pre>isheader() (translate.storage.subtitles.SubtitleUnit     method), 622 isheader() (translate.storage.tbx.tbxunit method),     627 isheader() (translate.storage.tiki.TikiUnit method),     633 isheader() (translate.storage.tmx.tmxunit method),</pre>
method), 642 isfuzzy() (translate.storage.ts2.tsunit method), 649 isfuzzy() (translate.storage.txt.TxtUnit method), 654 isfuzzy() (translate.storage.utx.UtxUnit method), 660 isfuzzy() (translate.storage.wordfast.WordfastUnit method), 670 isfuzzy() (translate.storage.xliff.xliffunit method),	isheader() (translate.storage.subtitles.SubtitleUnit method), 622 isheader() (translate.storage.tbx.tbxunit method), 627 isheader() (translate.storage.tiki.TikiUnit method), 633 isheader() (translate.storage.tmx.tmxunit method), 638 isheader() (translate.storage.trados.TradosUnit
method), 642 isfuzzy() (translate.storage.ts2.tsunit method), 649 isfuzzy() (translate.storage.txt.TxtUnit method), 654 isfuzzy() (translate.storage.utx.UtxUnit method), 660 isfuzzy() (translate.storage.wordfast.WordfastUnit method), 670 isfuzzy() (translate.storage.xliff.xliffunit method), 677 isheader() (translate.storage.base.TranslationUnit	<pre>isheader() (translate.storage.subtitles.SubtitleUnit     method), 622 isheader() (translate.storage.tbx.tbxunit method),     627 isheader() (translate.storage.tiki.TikiUnit method),     633 isheader() (translate.storage.tmx.tmxunit method),     638 isheader() (translate.storage.trados.TradosUnit     method), 642 isheader() (translate.storage.ts2.tsunit method), 649</pre>
method), 642 isfuzzy() (translate.storage.ts2.tsunit method), 649 isfuzzy() (translate.storage.txt.TxtUnit method), 654 isfuzzy() (translate.storage.utx.UtxUnit method), 660 isfuzzy() (translate.storage.wordfast.WordfastUnit method), 670 isfuzzy() (translate.storage.xliff.xliffunit method), 677	isheader() (translate.storage.subtitles.SubtitleUnit method), 622 isheader() (translate.storage.tbx.tbxunit method), 627 isheader() (translate.storage.tiki.TikiUnit method), 633 isheader() (translate.storage.tmx.tmxunit method), 638 isheader() (translate.storage.trados.TradosUnit method), 642
method), 642 isfuzzy() (translate.storage.ts2.tsunit method), 649 isfuzzy() (translate.storage.txt.TxtUnit method), 654 isfuzzy() (translate.storage.utx.UtxUnit method), 660 isfuzzy() (translate.storage.wordfast.WordfastUnit method), 670 isfuzzy() (translate.storage.xliff.xliffunit method), 677 isheader() (translate.storage.base.TranslationUnit method), 411 isheader() (translate.storage.catkeys.CatkeysUnit method), 418	<pre>isheader() (translate.storage.subtitles.SubtitleUnit</pre>
method), 642  isfuzzy() (translate.storage.ts2.tsunit method), 649  isfuzzy() (translate.storage.txt.TxtUnit method), 654  isfuzzy() (translate.storage.utx.UtxUnit method), 660  isfuzzy() (translate.storage.wordfast.WordfastUnit method), 670  isfuzzy() (translate.storage.xliff.xliffunit method), 677  isheader() (translate.storage.base.TranslationUnit method), 411  isheader() (translate.storage.catkeys.CatkeysUnit method), 418  isheader() (translate.storage.csvl10n.csvunit	isheader() (translate.storage.subtitles.SubtitleUnit method), 622 isheader() (translate.storage.tbx.tbxunit method), 627 isheader() (translate.storage.tiki.TikiUnit method), 633 isheader() (translate.storage.tmx.tmxunit method), 638 isheader() (translate.storage.trados.TradosUnit method), 642 isheader() (translate.storage.ts2.tsunit method), 649 isheader() (translate.storage.txt.TxtUnit method), 654 isheader() (translate.storage.utx.UtxUnit method), 660
method), 642 isfuzzy() (translate.storage.ts2.tsunit method), 649 isfuzzy() (translate.storage.txt.TxtUnit method), 654 isfuzzy() (translate.storage.utx.UtxUnit method), 660 isfuzzy() (translate.storage.wordfast.WordfastUnit method), 670 isfuzzy() (translate.storage.xliff.xliffunit method), 677 isheader() (translate.storage.base.TranslationUnit method), 411 isheader() (translate.storage.catkeys.CatkeysUnit method), 418 isheader() (translate.storage.csvl10n.csvunit method), 423	isheader() (translate.storage.subtitles.SubtitleUnit method), 622 isheader() (translate.storage.tbx.tbxunit method), 627 isheader() (translate.storage.tiki.TikiUnit method), 633 isheader() (translate.storage.tmx.tmxunit method), 638 isheader() (translate.storage.trados.TradosUnit method), 642 isheader() (translate.storage.ts2.tsunit method), 649 isheader() (translate.storage.txt.TxtUnit method), 654 isheader() (translate.storage.utx.UtxUnit method), 660 isheader() (translate.storage.wordfast.WordfastUnit
method), 642  isfuzzy() (translate.storage.ts2.tsunit method), 649  isfuzzy() (translate.storage.txt.TxtUnit method), 654  isfuzzy() (translate.storage.utx.UtxUnit method), 660  isfuzzy() (translate.storage.wordfast.WordfastUnit method), 670  isfuzzy() (translate.storage.xliff.xliffunit method), 677  isheader() (translate.storage.base.TranslationUnit method), 411  isheader() (translate.storage.catkeys.CatkeysUnit method), 418  isheader() (translate.storage.csvl10n.csvunit	isheader() (translate.storage.subtitles.SubtitleUnit method), 622 isheader() (translate.storage.tbx.tbxunit method), 627 isheader() (translate.storage.tiki.TikiUnit method), 633 isheader() (translate.storage.tmx.tmxunit method), 638 isheader() (translate.storage.trados.TradosUnit method), 642 isheader() (translate.storage.ts2.tsunit method), 649 isheader() (translate.storage.txt.TxtUnit method), 654 isheader() (translate.storage.utx.UtxUnit method), 660

isleaf() (translate.storage.placeables.base.Bpt method), 498	isleaf() (translate.storage.placeables.xliff.UnknownXML method), 546
isleaf() (translate.storage.placeables.base.Bx method), 506	isleaf() (translate.storage.placeables.xliff.X method), 534
isleaf() (translate.storage.placeables.base.Ept method), 500	islower() (translate.misc.multistring.multistring method), 388
isleaf() (translate.storage.placeables.base.Ex method), 508	isnull() (translate.storage.dtd.dtdunit method), 429 isnumeric() (translate.misc.multistring.multistring
isleaf() (translate.storage.placeables.base.G method), 505	method), 388 isobsolete() (trans-
isleaf() (translate.storage.placeables.base.It method), 503	late.storage.base.TranslationUnit method), 412
isleaf() (translate.storage.placeables.base.Ph method), 501	isobsolete() (translate.storage.catkeys.CatkeysUnit method), 418
isleaf() (translate.storage.placeables.base.Sub method), 511	isobsolete() (translate.storage.csvl10n.csvunit method), 423
isleaf() (translate.storage.placeables.base.X method), 509	
isleaf() (translate.storage.placeables.general.AltAttrF method), 513	
isleaf() (translate.storage.placeables.general.XMLEn. method), 515	
	gPlaceble lete() (translate.storage.ini.iniunit method), 449
isleaf() (translate.storage.placeables.interfaces.BaseAmethod), 518	Placeableolete() (trans- late.storage.jsonl10n.I18NextUnit method),
isleaf()(translate.storage.placeables.interfaces.Invision	blePlaceabl <del>4</del> 55
method), 520	isobsolete() (trans-
method), 521	ingPlaceabllate.storage.jsonl10n.JsonNestedUnit method), 461
method), 523	nceineantPladeentble() (translate.storage.jsonl10n.JsonUnit method), 465
isleaf() (translate.storage.placeables.interfaces.Subflo	
method), 525 isleaf() (translate.storage.placeables.strelem.StringEl	late.storage.jsonl10n.WebExtensionJsonUnit lem method), 469
method), 527 isleaf() (translate.storage.placeables.terminology.Te	isobsolete() (translate.storage.lisa.LISAunit
	minologyPlanethhd) 475
method), 529 isleaf() (translate.storage.placeables.xliff.Bpt	isobsolete() (translate.storage.mo.mounit method), 481
method), 529 isleaf() (translate.storage.placeables.xliff.Bpt method), 531	isobsolete() (translate.storage.mo.mounit method), 481 isobsolete() (trans-
method), 529 isleaf() (translate.storage.placeables.xliff.Bpt method), 531 isleaf() (translate.storage.placeables.xliff.Bx	isobsolete() (translate.storage.mo.mounit method), 481
method), 529 isleaf() (translate.storage.placeables.xliff.Bpt method), 531	isobsolete() (translate.storage.mo.mounit method), 481 isobsolete() (translate.storage.mozilla_lang.LangUnit method),
method), 529  isleaf() (translate.storage.placeables.xliff.Bpt method), 531  isleaf() (translate.storage.placeables.xliff.Bx method), 536  isleaf() (translate.storage.placeables.xliff.Ept method), 533	isobsolete() (translate.storage.mo.mounit method), 481 isobsolete() (translate.storage.mozilla_lang.LangUnit method), 486 isobsolete() (translate.storage.omegat.OmegaTUnit method),
method), 529  isleaf() (translate.storage.placeables.xliff.Bpt method), 531  isleaf() (translate.storage.placeables.xliff.Bx method), 536  isleaf() (translate.storage.placeables.xliff.Ept method), 533  isleaf() (translate.storage.placeables.xliff.Ex method), 538	isobsolete() (translate.storage.mo.mounit method), 481 isobsolete() (translate.storage.mozilla_lang.LangUnit method), 486 isobsolete() (translate.storage.omegat.OmegaTUnit method), 493 isobsolete() (translate.storage.php.phpunit
method), 529  isleaf() (translate.storage.placeables.xliff.Bpt method), 531  isleaf() (translate.storage.placeables.xliff.Bx method), 536  isleaf() (translate.storage.placeables.xliff.Ept method), 533  isleaf() (translate.storage.placeables.xliff.Ex	isobsolete() (translate.storage.mo.mounit method), 481 isobsolete() (translate.storage.mozilla_lang.LangUnit method), 486 isobsolete() (translate.storage.omegat.OmegaTUnit method), 493
method), 529  isleaf() (translate.storage.placeables.xliff.Bpt method), 531  isleaf() (translate.storage.placeables.xliff.Bx method), 536  isleaf() (translate.storage.placeables.xliff.Ept method), 533  isleaf() (translate.storage.placeables.xliff.Ex method), 538  isleaf() (translate.storage.placeables.xliff.G	isobsolete() (translate.storage.mo.mounit method), 481 isobsolete() (translate.storage.mozilla_lang.LangUnit method), 486 isobsolete() (translate.storage.omegat.OmegaTUnit method), 493 isobsolete() (translate.storage.php.phpunit method), 550 isobsolete() (translate.storage.pocommon.pounit method), 556
method), 529  isleaf() (translate.storage.placeables.xliff.Bpt method), 531  isleaf() (translate.storage.placeables.xliff.Bx method), 536  isleaf() (translate.storage.placeables.xliff.Ept method), 533  isleaf() (translate.storage.placeables.xliff.Ex method), 538  isleaf() (translate.storage.placeables.xliff.G method), 539  isleaf() (translate.storage.placeables.xliff.It method), 541  isleaf() (translate.storage.placeables.xliff.Ph	isobsolete() (translate.storage.mo.mounit method), 481 isobsolete() (translate.storage.mozilla_lang.LangUnit method), 486 isobsolete() (translate.storage.omegat.OmegaTUnit method), 493 isobsolete() (translate.storage.php.phpunit method), 550 isobsolete() (translate.storage.pocommon.pounit method), 556 isobsolete() (translate.storage.poxliff.PoXliffUnit method), 565
method), 529  isleaf() (translate.storage.placeables.xliff.Bpt method), 531  isleaf() (translate.storage.placeables.xliff.Bx method), 536  isleaf() (translate.storage.placeables.xliff.Ept method), 533  isleaf() (translate.storage.placeables.xliff.Ex method), 538  isleaf() (translate.storage.placeables.xliff.G method), 539  isleaf() (translate.storage.placeables.xliff.It method), 541	isobsolete() (translate.storage.mo.mounit method), 481 isobsolete() (translate.storage.mozilla_lang.LangUnit method), 486 isobsolete() (translate.storage.omegat.OmegaTUnit method), 493 isobsolete() (translate.storage.php.phpunit method), 550 isobsolete() (translate.storage.pocommon.pounit method), 556 isobsolete() (translate.storage.poxliff.PoXliffUnit

method), 593		late. tools. poconflicts. Conflict Option Parser		
isobsolete() (translate.storage.q	m.qmunit method),	method), 684		
598		isrecursiv	re()	(trans-
<pre>isobsolete() (translate.sta method), 604</pre>	orage.qph.QphUnit	late.t 688	ools.pogrep.GrepOptionParser	method),
isobsolete() (translate.storage.	rc.rcunit method),	isrecursiv	re()	(trans-
609	,,		ools.porestructure.SplitOptionP	arser
isobsolete()	(trans-		od), 691	
late.storage.subtitles.Subtitle	•	isrecursiv		(trans-
622	,,		ools.poterminology.Terminology	OptionParser
<pre>isobsolete() (translate.storage.th</pre>	ex.tbxunit method),		od), 695	•
628		isreview()	(translate.filters.checks.Standar	dUnitChecker
isobsolete() (translate.s	torage.tiki.TikiUnit	meth	od), 338	
method), 633		isreview()	(translate.storage.base.Trans	slationUnit
isobsolete() (translate.s	torage.tmx.tmxunit	meth	od), 412	
method), 638	-	isreview()	(translate.storage.catkeys.C	atkeysUnit
isobsolete() (translate.storage	.trados.TradosUnit	meth	od), 418	
method), 642		isreview()	(translate.storage.csvl1	On.csvunit
<pre>isobsolete() (translate.storage.t</pre>	s2.tsunit method),	meth	od), 423	
649		isreview()	(translate.storage.dtd.dtdunit	method),
<pre>isobsolete() (translate.storage.tx</pre>	t.TxtUnit method),	429		
654	•	isreview()	(translate.storage.html.htmluni	t method),
isobsolete() (translate.s	torage.utx.UtxUnit	439	·	,
method), 660		isreview()	(translate.storage.ical.icalunit	method),
isobsolete()	(trans-	444	·	
late.storage.wordfast.Wordfa	stUnit method),	isreview()	(translate.storage.ini.iniunit me	ethod), 449
670			(translate.storage.jsonl10n.II	
isobsolete() (translate.sa	torage.xliff.xliffunit		od), 455	
method), 677		isreview()	(translate.storage.jsonl10n.Json	าNestedUnit
isocode () (in module translate.lang	g.poedit), 377		od), 461	
ispurepunctuation() (in	module trans-	isreview()	(translate.storage.jsonl10	n.JsonUnit
late.filters.decoration), 347		meth	od), 465	
isrecursive()	(trans-	isreview()	(translate.storage.jsonl10n.Web	ExtensionJsonUnit
late.convert.convert.Archive	ConvertOptionParse	r meth	od), 469	
method), 235		isreview()	(translate.storage.lisa.LISAuni	t method),
isrecursive()	(trans-	475		
late.convert.convert.Convert	OptionParser	isreview()	(translate.storage.mo.mounit	method),
method), 238		481		
isrecursive()	(trans-	isreview()	(translate.storage.mozilla_lang	.LangUnit
late.convert.po2moz.MozCor	wertOptionParser	meth	od), 486	
method), 250		isreview()	(translate.storage.omegat.Or	negaTUnit
isrecursive()	(trans-	meth	od), 493	
late.convert.po2tmx.TmxOpt	ionParser	isreview()	(translate.storage.php.phpunit	method),
method), 255		550		
isrecursive()	(trans-	isreview()	(translate.storage.pocomn	non.pounit
late.convert.po2wordfast.Wf	OptionParser	meth	od), 556	
method), 260		isreview()	(translate.storage.poxliff.l	PoXliffUnit
isrecursive()	(trans-	meth	od), 565	
late.filters.pofilter.FilterOptic	onParser method),	isreview()	(translate.storage.propertie	es.propunit
349			od), 583	
isrecursive()	(trans-	isreview()	(translate.storage.pypo.pounit	method),
late.misc.optrecurse.Recursi	veOptionParser	593		
method), 391		isreview()	(translate.storage.qm.qmunit	method),
isrecursive()	(trans-	598		

$\verb isreview()  (translate.storage.qph.QphUnit method),$	istranslatable() (trans-
604	late. storage. js on 110 n. Web Extension Js on Unit
isreview() (translate.storage.rc.rcunit method), 609	method), 469
$\verb isreview()  (translate.storage.subtitles.SubtitleUnit $	istranslatable() (translate.storage.lisa.LISAunit
method), 622	method), 475
isreview() (translate.storage.tbx.tbxunit method),	istranslatable() (translate.storage.mo.mounit
628	method), 481
isreview() (translate.storage.tiki.TikiUnit method),	istranslatable() (trans-
633	late.storage.mozilla_lang.LangUnit method),
isreview() (translate.storage.tmx.tmxunit method), 638	486
	istranslatable() (trans-
isreview() (translate.storage.trados.TradosUnit method), 642	late.storage.omegat.OmegaTUnit method), 493
isreview() (translate.storage.ts2.tsunit method), 649	istranslatable() (translate.storage.php.phpunit
isreview() (translate.storage.txt.TxtUnit method),	method), 550
654	istranslatable() (trans-
isreview() (translate.storage.utx.UtxUnit method),	late.storage.pocommon.pounit method),
660	556
<pre>isreview() (translate.storage.wordfast.WordfastUnit</pre>	istranslatable() (trans-
method), 670	late.storage.poxliff.PoXliffUnit method),
isreview() (translate.storage.xliff.xliffunit method),	565
677	istranslatable() (trans-
isspace() (translate.misc.multistring.multistring	late.storage.properties.propunit method),
method), 388	583
<pre>istitle() (translate.misc.multistring.multistring</pre>	istranslatable() (translate.storage.pypo.pounit
method), 388	method), 593
istranslatable (trans-	istranslatable() (translate.storage.qm.qmunit
late.storage.placeables.strelem.StringElem	method), 599
attribute), 527	istranslatable() (translate.storage.qph.QphUnit
istranslatable() (trans-	method), 604
late.storage.base.TranslationUnit method),	istranslatable() (translate.storage.rc.rcunit
412	method), 609
istranslatable() (trans-	istranslatable() (trans-
late.storage.catkeys.CatkeysUnit method),	$late. storage. subtitles. Subtitle Unit \\ method),$
418	622
istranslatable() (trans-	<pre>istranslatable() (translate.storage.tbx.tbxunit</pre>
late.storage.csvl10n.csvunit method), 423	method), 628
istranslatable() (translate.storage.dtd.dtdunit	
method), 429	method), 633
<pre>istranslatable() (translate.storage.html.htmlunit</pre>	<pre>istranslatable() (translate.storage.tmx.tmxunit</pre>
method), 439	method), 638
istranslatable() (translate.storage.ical.icalunit	istranslatable() (trans-
method), 444	late.storage.trados.TradosUnit method),
istranslatable() (translate.storage.ini.iniunit	642
method), 449	istranslatable() (translate.storage.ts2.tsunit
istranslatable() (trans-	method), 649
late.storage.jsonl10n.I18NextUnit method),	istranslatable() (translate.storage.txt.TxtUnit
455	method), 654
istranslatable() (trans-	istranslatable() (translate.storage.utx.UtxUnit
late.storage.jsonl10n.JsonNestedUnit method), 462	method), 660 istranslatable() (trans-
istranslatable() (trans-	late.storage.wordfast.WordfastUnit method),
late.storage.jsonl10n.JsonUnit method),	670
465	istranslatable() (translate.storage.xliff.xliffunit

method), 677		istranslated()	(translate.storage.qph.QphU	Init
istranslated()	(trans-	method), 604		
late.storage.base.TranslationUnit n 412	nethod),	<pre>istranslated()     method), 609</pre>	(translate.storage.rc.rcı	ınit
istranslated()	(trans-	istranslated()	(tra	ıns-
late.storage.catkeys.CatkeysUnit n 418	nethod),	late.storage.si 623	ubtitles.SubtitleUnit metho	od),
istranslated() (translate.storage.csvl10n method), 423	a.csvunit	istranslated()  method), 628	(translate.storage.tbx.tbx)	ınit
istranslated() (translate.storage.dtd method), 430	l.dtdunit	istranslated()  method), 633	(translate.storage.tiki.TikiU	Init
istranslated() (translate.storage.html.method), 439	htmlunit	istranslated()  method), 638	(translate.storage.tmx.tmxi	ınit
istranslated() (translate.storage.ical. method), 444	icalunit.	istranslated()	(tra ados.TradosUnit metho	
istranslated() (translate.storage.in	i.iniunit	642		
<pre>method), 449 istranslated()</pre>	(trans-	<pre>istranslated()     method), 649</pre>	(translate.storage.ts2.tsu	ınıt
late.storage.jsonl10n.I18NextUnit n 455	nethod),	<pre>istranslated()     method), 655</pre>	(translate.storage.txt.TxtU	Init
<pre>istranslated()     late.storage.jsonl10n.JsonNestedUnit n</pre>	(trans- nethod),	istranslated()  method), 660	(translate.storage.utx.UtxU	
462 istranslated()	(trans-	istranslated()	tra) ordfast.WordfastUnit metho	
	nethod),	671	orajasi.worajasi0nii meino	π),
465		istranslated()	(translate.storage.xliff.xliffi	ınit
istranslated()	(trans-	method), 677		
late.storage.jsonl10n.WebExtensionJson method), 470	nUnit	isupper() (tran method), 388	slate.misc.multistring.multistr	ıng
<pre>istranslated() (translate.storage.lisa.L</pre>	LISAunit	isvalidaccelerat late.filters.dec		ıns-
<pre>istranslated() (translate.storage.mo method), 481</pre>	o.mounit	isvalidinputname late.convert.co	() (tra onvert.ArchiveConvertOptionP	
istranslated()	(trans-	method), 235	•	
late.storage.mozilla_lang.LangUnit n 486	nethod),	isvalidinputname <i>late.convert.co</i>	() (tra onvert.ConvertOptionParser	ıns-
istranslated()	(trans-	method), 238		
ŭ ŭ ŭ	nethod),	isvalidinputname		ıns-
493 istranslated() (translate.storage.php.	.phpunit	late.convert.po method), 250	o2moz.MozConvertOptionPars	ser
method), 551		isvalidinputname	() (tra	ıns-
istranslated()	(trans-	_	o2tmx.TmxOptionParser	
	nethod),	method), 256		
556		isvalidinputname		ıns-
<pre>istranslated()</pre>	(trans-	_	o2wordfast.WfOptionParser	
tate.storage.poxttyj.PoxttyjOnti n 566	nethod),	<i>method</i> ), 260 isvalidinputname	() (tra	111 C
istranslated()	(trans-	<del>-</del>	ilter.FilterOptionParser metho	
late.storage.properties.propunit n	nethod),	349	_	
583		isvalidinputname		
istranslated() (translate.storage.pype method), 593	Î	method), 391	ecurse.RecursiveOptionParser	
istranslated() (translate.storage.qm method), 599	ı.qmunit	isvalidinputname <i>late.tools.pocc</i>	() (tra onflicts.ConflictOptionParser	ıns-

method), 684		<pre>iter_depth_first()</pre>	(trans-
isvalidinputname()	(trans-	late.storage.placeables.general.XM	`
late.tools.pogrep.GrepOptionParser	method),	method), 516	
688		<pre>iter_depth_first()</pre>	(trans-
isvalidinputname()	(trans-	late.storage.placeables.interfaces.B	asePlaceable
late.tools.porestructure.SplitOption method), 691	rarser	<pre>method), 518 iter_depth_first()</pre>	(trans-
isvalidinputname()	(trans-	late.storage.placeables.interfaces.In	`
late.tools.poterminology.Terminolog	`		ivisible raceable
method), 695	,, - <sub>I</sub>	<pre>iter_depth_first()</pre>	(trans-
isvisible (translate.storage.placeables.str attribute), 527	relem.StringE		MaskingPlaceable
It (class in translate.storage.placeables.base	502	<pre>iter_depth_first()</pre>	(trans-
It (class in translate.storage.placeables.xliff)		late.storage.placeables.interfaces.R	`
it2po() (in module translate.convert.mozfi		method), 523	epiacement taccaste
244	V 1 1//	<pre>iter_depth_first()</pre>	(trans-
it2prop() (in module	trans-	late.storage.placeables.interfaces.S	ubflowPlaceable
late.convert.mozfunny2prop), 244		method), 525	
items()(translate.misc.dictutils.cidict meth		<pre>iter_depth_first()</pre>	(trans-
items() (translate.storage.oo.unon method), 496	rmalizechar	late.storage.placeables.strelem.Stri method), 527	ngElem
iter_depth_first()	(trans-	iter_depth_first()	(trans-
late.storage.placeables.base.Bpt	method),	late.storage.placeables.terminology	`
498	,,	method), 529	07
<pre>iter_depth_first()</pre>	(trans-	iter_depth_first()	(trans-
late.storage.placeables.base.Bx 506	method),	late.storage.placeables.xliff.Bpt 531	method),
<pre>iter_depth_first()</pre>	(trans-	<pre>iter_depth_first()</pre>	(trans-
	(ii aiis	icci_acpcii_iiisc()	(VI CVI VS
late.storage.placeables.base.Ept	method),	late.storage.placeables.xliff.Bx	method),
late.storage.placeables.base.Ept 500		late.storage.placeables.xliff.Bx 536	`
<pre>late.storage.placeables.base.Ept 500 iter_depth_first()</pre>	method), (trans-	<pre>late.storage.placeables.xliff.Bx 536 iter_depth_first()</pre>	method), (trans-
<pre>late.storage.placeables.base.Ept 500 iter_depth_first() late.storage.placeables.base.Ex</pre>	method),	<pre>late.storage.placeables.xliff.Bx 536 iter_depth_first() late.storage.placeables.xliff.Ept</pre>	method),
<pre>late.storage.placeables.base.Ept 500 iter_depth_first()     late.storage.placeables.base.Ex 508</pre>	method), (trans- method),	<pre>late.storage.placeables.xliff.Bx 536 iter_depth_first() late.storage.placeables.xliff.Ept 533</pre>	method), (trans- method),
<pre>late.storage.placeables.base.Ept 500 iter_depth_first()     late.storage.placeables.base.Ex 508 iter_depth_first()</pre>	method),  (trans- method),  (trans-	<pre>late.storage.placeables.xliff.Bx 536 iter_depth_first()     late.storage.placeables.xliff.Ept 533 iter_depth_first()</pre>	method),  (trans- method),  (trans-
<pre>late.storage.placeables.base.Ept 500 iter_depth_first()     late.storage.placeables.base.Ex 508</pre>	method), (trans- method),	<pre>late.storage.placeables.xliff.Bx 536 iter_depth_first() late.storage.placeables.xliff.Ept 533</pre>	method), (trans- method),
<pre>late.storage.placeables.base.Ept 500 iter_depth_first()     late.storage.placeables.base.Ex 508 iter_depth_first()     late.storage.placeables.base.G</pre>	method),  (trans- method),  (trans-	<pre>late.storage.placeables.xliff.Bx 536 iter_depth_first()     late.storage.placeables.xliff.Ept 533 iter_depth_first()     late.storage.placeables.xliff.Ex</pre>	method),  (trans- method),  (trans-
<pre>late.storage.placeables.base.Ept 500 iter_depth_first()     late.storage.placeables.base.Ex 508 iter_depth_first()     late.storage.placeables.base.G 505 iter_depth_first()     late.storage.placeables.base.It</pre>	method),  (trans- method),  (trans- method),	<pre>late.storage.placeables.xliff.Bx 536 iter_depth_first()     late.storage.placeables.xliff.Ept 533 iter_depth_first()     late.storage.placeables.xliff.Ex 538 iter_depth_first()     late.storage.placeables.xliff.G meth</pre>	method),  (trans- method),  (trans- method),  (trans- od), 539
<pre>late.storage.placeables.base.Ept 500 iter_depth_first()     late.storage.placeables.base.Ex 508 iter_depth_first()     late.storage.placeables.base.G 505 iter_depth_first()     late.storage.placeables.base.It 503</pre>	method),  (trans- method),  (trans- method),  (trans- method),	<pre>late.storage.placeables.xliff.Bx 536 iter_depth_first()     late.storage.placeables.xliff.Ept 533 iter_depth_first()     late.storage.placeables.xliff.Ex 538 iter_depth_first()     late.storage.placeables.xliff.G meth iter_depth_first()</pre>	method),  (trans- method),  (trans- method),  (trans- od), 539  (trans-
<pre>late.storage.placeables.base.Ept 500 iter_depth_first()     late.storage.placeables.base.Ex 508 iter_depth_first()     late.storage.placeables.base.G 505 iter_depth_first()     late.storage.placeables.base.It 503 iter_depth_first()</pre>	method),  (trans- method),  (trans- method),  (trans- method),  (trans-	<pre>late.storage.placeables.xliff.Bx 536 iter_depth_first()     late.storage.placeables.xliff.Ept 533 iter_depth_first()     late.storage.placeables.xliff.Ex 538 iter_depth_first()     late.storage.placeables.xliff.G meth iter_depth_first()     late.storage.placeables.xliff.It methe</pre>	method),  (trans- method),  (trans- method),  (trans- od), 539  (trans- od), 541
<pre>late.storage.placeables.base.Ept 500 iter_depth_first()     late.storage.placeables.base.Ex 508 iter_depth_first()     late.storage.placeables.base.G 505 iter_depth_first()     late.storage.placeables.base.It 503 iter_depth_first()     late.storage.placeables.base.Ph</pre>	method),  (trans- method),  (trans- method),  (trans- method),	<pre>late.storage.placeables.xliff.Bx 536 iter_depth_first()     late.storage.placeables.xliff.Ept 533 iter_depth_first()     late.storage.placeables.xliff.Ex 538 iter_depth_first()     late.storage.placeables.xliff.G meth iter_depth_first()     late.storage.placeables.xliff.It method iter_depth_first()</pre>	method),  (trans- method),  (trans- method),  (trans- od), 539  (trans- od), 541  (trans-
late.storage.placeables.base.Ept 500 iter_depth_first()     late.storage.placeables.base.Ex 508 iter_depth_first()     late.storage.placeables.base.G 505 iter_depth_first()     late.storage.placeables.base.It 503 iter_depth_first()     late.storage.placeables.base.Ph 502	method),  (trans- method),  (trans- method),  (trans- method),  (trans- method),	late.storage.placeables.xliff.Bx 536 iter_depth_first()     late.storage.placeables.xliff.Ept 533 iter_depth_first()     late.storage.placeables.xliff.Ex 538 iter_depth_first()     late.storage.placeables.xliff.G meth iter_depth_first()     late.storage.placeables.xliff.It meth iter_depth_first()     late.storage.placeables.xliff.It meth iter_depth_first()     late.storage.placeables.xliff.Ph	method),  (trans- method),  (trans- method),  (trans- od), 539  (trans- od), 541
<pre>late.storage.placeables.base.Ept 500 iter_depth_first()     late.storage.placeables.base.Ex 508 iter_depth_first()     late.storage.placeables.base.G 505 iter_depth_first()     late.storage.placeables.base.It 503 iter_depth_first()     late.storage.placeables.base.Ph 502 iter_depth_first()</pre>	method),  (trans- method),  (trans- method),  (trans- method),  (trans- method),  (trans- method),	late.storage.placeables.xliff.Bx 536 iter_depth_first()     late.storage.placeables.xliff.Ept 533 iter_depth_first()     late.storage.placeables.xliff.Ex 538 iter_depth_first()     late.storage.placeables.xliff.G meth iter_depth_first()     late.storage.placeables.xliff.It meth iter_depth_first()     late.storage.placeables.xliff.It meth iter_depth_first()     late.storage.placeables.xliff.Ph 544	method),  (trans- method),  (trans- method),  (trans- od), 539  (trans- od), 541  (trans- method),
late.storage.placeables.base.Ept 500 iter_depth_first()     late.storage.placeables.base.Ex 508 iter_depth_first()     late.storage.placeables.base.G 505 iter_depth_first()     late.storage.placeables.base.It 503 iter_depth_first()     late.storage.placeables.base.Ph 502	method),  (trans- method),  (trans- method),  (trans- method),  (trans- method),	late.storage.placeables.xliff.Bx 536 iter_depth_first()     late.storage.placeables.xliff.Ept 533 iter_depth_first()     late.storage.placeables.xliff.Ex 538 iter_depth_first()     late.storage.placeables.xliff.G meth iter_depth_first()     late.storage.placeables.xliff.It meth iter_depth_first()     late.storage.placeables.xliff.It meth iter_depth_first()     late.storage.placeables.xliff.Ph	method),  (trans- method),  (trans- method),  (trans- od), 539  (trans- od), 541  (trans-
<pre>late.storage.placeables.base.Ept 500 iter_depth_first()     late.storage.placeables.base.Ex 508 iter_depth_first()     late.storage.placeables.base.G 505 iter_depth_first()     late.storage.placeables.base.It 503 iter_depth_first()     late.storage.placeables.base.Ph 502 iter_depth_first()     late.storage.placeables.base.Sub</pre>	method),  (trans- method),  (trans- method),  (trans- method),  (trans- method),  (trans- method),	<pre>late.storage.placeables.xliff.Bx 536 iter_depth_first()     late.storage.placeables.xliff.Ept 533 iter_depth_first()     late.storage.placeables.xliff.Ex 538 iter_depth_first()     late.storage.placeables.xliff.G meth iter_depth_first()     late.storage.placeables.xliff.It method iter_depth_first()     late.storage.placeables.xliff.It method iter_depth_first()     late.storage.placeables.xliff.Ph 544 iter_depth_first()</pre>	method),  (trans- method),  (trans- method),  (trans- od), 539  (trans- od), 541  (trans- method),  (trans- method),  (trans-
<pre>late.storage.placeables.base.Ept 500 iter_depth_first()     late.storage.placeables.base.Ex 508 iter_depth_first()     late.storage.placeables.base.G 505 iter_depth_first()     late.storage.placeables.base.It 503 iter_depth_first()     late.storage.placeables.base.Ph 502 iter_depth_first()     late.storage.placeables.base.Sub 511 iter_depth_first()     late.storage.placeables.base.X</pre>	method),  (trans- method),  (trans- method),  (trans- method),  (trans- method),  (trans- method),	<pre>late.storage.placeables.xliff.Bx 536 iter_depth_first()     late.storage.placeables.xliff.Ept 533 iter_depth_first()     late.storage.placeables.xliff.Ex 538 iter_depth_first()     late.storage.placeables.xliff.G meth iter_depth_first()     late.storage.placeables.xliff.It method iter_depth_first()     late.storage.placeables.xliff.Ph 544 iter_depth_first()     late.storage.placeables.xliff.Sub 543 iter_depth_first()</pre>	method),  (trans- method),  (trans- method),  (trans- od), 539  (trans- od), 541  (trans- method),  (trans- method),  (trans- method),
late.storage.placeables.base.Ept 500  iter_depth_first()     late.storage.placeables.base.Ex 508  iter_depth_first()     late.storage.placeables.base.G 505  iter_depth_first()     late.storage.placeables.base.It 503  iter_depth_first()     late.storage.placeables.base.Ph 502  iter_depth_first()     late.storage.placeables.base.Sub 511  iter_depth_first()     late.storage.placeables.base.Sub 511  iter_depth_first()     late.storage.placeables.base.X 510	method),  (trans- method),  (trans- method),  (trans- method),  (trans- method),  (trans- method),  (trans- method),	late.storage.placeables.xliff.Bx 536  iter_depth_first()     late.storage.placeables.xliff.Ept 533  iter_depth_first()     late.storage.placeables.xliff.Ex 538  iter_depth_first()     late.storage.placeables.xliff.G meth iter_depth_first()     late.storage.placeables.xliff.It method iter_depth_first()     late.storage.placeables.xliff.Ph 544  iter_depth_first()     late.storage.placeables.xliff.Sub 543  iter_depth_first()     late.storage.placeables.xliff.Sub	method),  (trans- method),  (trans- method),  (trans- od), 539  (trans- od), 541  (trans- method),  (trans- method),  (trans- method),
<pre>late.storage.placeables.base.Ept 500 iter_depth_first()     late.storage.placeables.base.Ex 508 iter_depth_first()     late.storage.placeables.base.G 505 iter_depth_first()     late.storage.placeables.base.It 503 iter_depth_first()     late.storage.placeables.base.Ph 502 iter_depth_first()     late.storage.placeables.base.Sub 511 iter_depth_first()     late.storage.placeables.base.Sub 510 iter_depth_first()     late.storage.placeables.base.X 510 iter_depth_first()</pre>	method),  (trans- method),	late.storage.placeables.xliff.Bx 536  iter_depth_first()     late.storage.placeables.xliff.Ept 533  iter_depth_first()     late.storage.placeables.xliff.Ex 538  iter_depth_first()     late.storage.placeables.xliff.G meth iter_depth_first()     late.storage.placeables.xliff.It meth iter_depth_first()     late.storage.placeables.xliff.Ph 544  iter_depth_first()     late.storage.placeables.xliff.Sub 543  iter_depth_first()     late.storage.placeables.xliff.Sub 543  iter_depth_first()     late.storage.placeables.xliff.Unknow method), 546	method),  (trans- method),  (trans- method),  (trans- od), 539  (trans- od), 541  (trans- method),  (trans- method),  (trans- method),  (trans- method),
late.storage.placeables.base.Ept 500  iter_depth_first()     late.storage.placeables.base.Ex 508  iter_depth_first()     late.storage.placeables.base.G 505  iter_depth_first()     late.storage.placeables.base.It 503  iter_depth_first()     late.storage.placeables.base.Ph 502  iter_depth_first()     late.storage.placeables.base.Sub 511  iter_depth_first()     late.storage.placeables.base.X 510  iter_depth_first()     late.storage.placeables.base.X 510  iter_depth_first()     late.storage.placeables.general.AltA	method),  (trans- method),	late.storage.placeables.xliff.Bx 536  iter_depth_first()     late.storage.placeables.xliff.Ept 533  iter_depth_first()     late.storage.placeables.xliff.Ex 538  iter_depth_first()     late.storage.placeables.xliff.G meth iter_depth_first()     late.storage.placeables.xliff.It method iter_depth_first()     late.storage.placeables.xliff.Ph 544  iter_depth_first()     late.storage.placeables.xliff.Sub 543  iter_depth_first()     late.storage.placeables.xliff.Unknown method), 546  iter_depth_first()	method),  (trans- method),  (trans- method),  (trans- od), 539  (trans- od), 541  (trans- method),  (trans- method),
late.storage.placeables.base.Ept 500  iter_depth_first()     late.storage.placeables.base.Ex 508  iter_depth_first()     late.storage.placeables.base.G 505  iter_depth_first()     late.storage.placeables.base.It 503  iter_depth_first()     late.storage.placeables.base.Ph 502  iter_depth_first()     late.storage.placeables.base.Sub 511  iter_depth_first()     late.storage.placeables.base.X 510  iter_depth_first()     late.storage.placeables.base.X 510  iter_depth_first()     late.storage.placeables.general.AltA method), 513	method),  (trans- method),	late.storage.placeables.xliff.Bx 536  iter_depth_first()     late.storage.placeables.xliff.Ept 533  iter_depth_first()     late.storage.placeables.xliff.Ex 538  iter_depth_first()     late.storage.placeables.xliff.G meth iter_depth_first()     late.storage.placeables.xliff.It meth iter_depth_first()     late.storage.placeables.xliff.Ph 544  iter_depth_first()     late.storage.placeables.xliff.Sub 543  iter_depth_first()     late.storage.placeables.xliff.Unknow     method), 546  iter_depth_first()     late.storage.placeables.xliff.Y method iter_depth_first()     late.storage.placeables.xliff.Y method	method),  (trans- method),  (trans- method),  (trans- od), 539  (trans- od), 541  (trans- method),  (trans- method),  (trans- method),  (trans- method),  (trans- method),  (trans- method),
late.storage.placeables.base.Ept 500  iter_depth_first()     late.storage.placeables.base.Ex 508  iter_depth_first()     late.storage.placeables.base.G 505  iter_depth_first()     late.storage.placeables.base.It 503  iter_depth_first()     late.storage.placeables.base.Ph 502  iter_depth_first()     late.storage.placeables.base.Sub 511  iter_depth_first()     late.storage.placeables.base.X 510  iter_depth_first()     late.storage.placeables.base.X 510  iter_depth_first()     late.storage.placeables.general.AltA     method), 513  iter_depth_first()	method),  (trans- method),	late.storage.placeables.xliff.Bx 536  iter_depth_first()     late.storage.placeables.xliff.Ept 533  iter_depth_first()     late.storage.placeables.xliff.Ex 538  iter_depth_first()     late.storage.placeables.xliff.G meth iter_depth_first()     late.storage.placeables.xliff.It meth iter_depth_first()     late.storage.placeables.xliff.Ph 544  iter_depth_first()     late.storage.placeables.xliff.Sub 543  iter_depth_first()     late.storage.placeables.xliff.Unknow     method), 546  iter_depth_first()     late.storage.placeables.xliff.X method iter_depth_first()     late.storage.placeables.xliff.X method iter_depth_first()     late.storage.placeables.xliff.X method iter_items() (translate.misc.dictutils.cidit	method),  (trans- method),  (trans- method),  (trans- od), 539  (trans- od), 541  (trans- method),  (trans- method),  (trans- method),  (trans- method),  (trans- method),  (trans- method),
late.storage.placeables.base.Ept 500  iter_depth_first()     late.storage.placeables.base.Ex 508  iter_depth_first()     late.storage.placeables.base.G 505  iter_depth_first()     late.storage.placeables.base.It 503  iter_depth_first()     late.storage.placeables.base.Ph 502  iter_depth_first()     late.storage.placeables.base.Sub 511  iter_depth_first()     late.storage.placeables.base.X 510  iter_depth_first()     late.storage.placeables.base.X 510  iter_depth_first()     late.storage.placeables.general.AltA method), 513	method),  (trans- method),	late.storage.placeables.xliff.Bx 536  iter_depth_first()     late.storage.placeables.xliff.Ept 533  iter_depth_first()     late.storage.placeables.xliff.Ex 538  iter_depth_first()     late.storage.placeables.xliff.G meth iter_depth_first()     late.storage.placeables.xliff.It meth iter_depth_first()     late.storage.placeables.xliff.Ph 544  iter_depth_first()     late.storage.placeables.xliff.Sub 543  iter_depth_first()     late.storage.placeables.xliff.Unknow     method), 546  iter_depth_first()     late.storage.placeables.xliff.X method iter_depth_first()     late.storage.placeables.xliff.X method iter_depth_first()     late.storage.placeables.xliff.X method iter_items() (translate.misc.dictutils.cidit	method),  (trans- method),  (trans- method),  (trans- od), 539  (trans- od), 541  (trans- method),

method), 496	kdecomments() (trans-
iterkeys() (translate.misc.dictutils.cidict method), 386	late.filters.checks.MinimalChecker method), 311
iterkeys () (translate.storage.oo.unormalizechar method), 496	kdecomments() (trans- late.filters.checks.MozillaChecker method),
itervaluerefs() (trans-	317
late.misc.lru.LRUCachingDict method), 387	kdecomments() (trans- late.filters.checks.OpenOfficeChecker method),
itervalues() (translate.misc.dictutils.cidict method), 386	323 kdecomments() (trans-
itervalues() (translate.storage.oo.unormalizechar method), 496	late.filters.checks.ReducedChecker method), 329
J	kdecomments() (trans- late.filters.checks.StandardChecker method),
ja ( <i>class in translate.lang.ja</i> ), 371	334
<pre>java_utf8_properties_encode() (in module</pre>	kdecomments() (translate.filters.checks.TermChecker method), 342
javafile (class in translate.storage.properties), 575 javapropertiesencode() (in module trans-	key_strip() (translate.storage.properties.Dialect class method), 570
late.misc.quote), 395	key_strip() (trans-
javautf8file (class in translate.storage.properties), 576	late.storage.properties.DialectFlex class method), 571
join() (translate.misc.multistring.multistring method),	key_strip() (trans- late.storage.properties.DialectGaia class
388 joomlafile (class in translate.storage.properties),	late.storage.properties.DialectGaia class method), 571
578	key_strip() (trans-
json2po (class in translate.convert.json2po), 243 JsonFile (class in translate.storage.json110n), 456	late.storage.properties.DialectJava class method), 571
JsonNestedFile (class in trans-	key_strip() (trans-
late.storage.jsonl10n), 458  JsonNestedUnit (class in trans-	late.storage.properties.DialectJavaUtf8 class method), 572
late.storage.jsonl10n), 460	key_strip() (trans-
JsonUnit (class in translate.storage.jsonl10n), 463	late.storage.properties.DialectJoomla class method), 572
K	key_strip() (trans-
KdeChecker (class in translate.filters.checks), 292 kdecomments () (trans-	late.storage.properties.DialectMozilla class method), 573
late.filters.checks.CCLicenseChecker method),	key_strip() (trans-
271	late.storage.properties.DialectSkype class method), 573
kdecomments() (trans-	key_strip() (trans-
late.filters.checks.DrupalChecker method), 277	late.storage.properties.DialectStrings class method), 574
kdecomments() (trans- late.filters.checks.GnomeChecker method),	key_strip() (trans-
283	late.storage.properties.DialectStringsUtf8 class method), 574
kdecomments() (translate.filters.checks.IOSChecker method), 288	keys () (translate.misc.dictutils.cidict method), 386
kdecomments () (translate.filters.checks.KdeChecker method), 294	keys () (translate.storage.oo.unormalizechar method), 496
kdecomments() (translate.filters.checks.L20nChecker method), 300	khmerpunc (translate.lang.km.km attribute), 372 km (class in translate.lang.km), 372
kdecomments() (trans-	kn (class in translate.lang.kn), 372
late.filters.checks.LibreOfficeChecker method), 306	ko (class in translate.lang.ko), 373

L	length_difference() (translate.lang.ja.ja class
L20nChecker (class in translate.filters.checks), 298	method), 371
labelsuffixes (in module translate.storage.dtd), 431	<pre>length_difference() (translate.lang.km.km class</pre>
labelsuffixes (in module trans-	<pre>length_difference() (translate.lang.kn.kn class     method), 373</pre>
late.storage.properties), 580	length_difference() (translate.lang.ko.ko class
lang2po (class in translate.convert.mozlang2po), 244	method), 373
lang_codes (in module translate.lang.poedit), 378	length_difference() (translate.lang.ml.ml class
lang_names (in module translate.lang.poedit), 378	method), 374
LANG_TEAM_CONTACT_SNIPPETS (in module translate.lang.team), 381	length_difference() (translate.lang.mr.mr class
LangStore (class in translate.storage.mozilla_lang),	method), 375
482	<pre>length_difference() (translate.lang.ne.ne class</pre>
LanguageError, 634	method), 376
languagematch() (in module translate.lang.data),	<pre>length_difference() (translate.lang.pa.pa class</pre>
360	method), 377
languages (in module translate.lang.data), 360	<pre>length_difference() (translate.lang.si.si class</pre>
LangUnit (class in translate.storage.mozilla_lang), 484	method), 378
<pre>launch_server() (in module translate.misc.wsgi), 395</pre>	<pre>length_difference() (translate.lang.st.st class     method), 379</pre>
<pre>length_difference() (translate.lang.af.af class</pre>	<pre>length_difference() (translate.lang.sv.sv class</pre>
method), 353	method), 380
<pre>length_difference() (translate.lang.am.am class</pre>	<pre>length_difference() (translate.lang.ta.ta class</pre>
method), 354	method), 380
<pre>length_difference() (translate.lang.ar.ar class</pre>	length_difference() (translate.lang.te.te class
method), 354	method), 381
$\verb length_difference()  (\textit{translate.lang.bn.bn} \ \textit{class}$	length_difference() (translate.lang.th.th class
method), 355	method), 382
<pre>length_difference() (trans-</pre>	<pre>length_difference() (translate.lang.ug.ug class     method), 383</pre>
late.lang.code_or.code_or class method),	length_difference() (translate.lang.ur.ur class
356	method), 384
<pre>length_difference() (trans- late.lang.common.Common class method),</pre>	length_difference() (translate.lang.vi.vi class
358	method), 385
<pre>length_difference() (translate.lang.de.de class</pre>	LibreOfficeChecker (class in trans-
method), 363	late.filters.checks), 304
<pre>length_difference() (translate.lang.el.el class</pre>	LISAfile (class in translate.storage.lisa), 471
method), 364	LISAunit (class in translate.storage.lisa), 473 listseperator (translate.lang.common.Common at-
<pre>length_difference() (translate.lang.es.es class     method), 365</pre>	tribute), 358
<pre>length_difference() (translate.lang.fa.fa class</pre>	listsubfiles() (translate.storage.oo.oomultifile
method), 366	method), 495
<pre>length_difference() (translate.lang.fi.fi class</pre>	ljust() (translate.misc.multistring.multistring
method), 366	method), 389
<pre>length_difference() (translate.lang.fr.fr class</pre>	load() (translate.storage.bundleprojstore.BundleProjectStore
method), 367	method), 414
<pre>length_difference() (translate.lang.gu.gu class</pre>	load() (translate.storage.projstore.ProjectStore
method), 368	method), 568 long() (translate.filters.checks.CCLicenseChecker
length_difference() (translate.lang.he.he class	long () (translate.filters.checks.CCLicenseChecker method), 271
method), 369	long() (translate.filters.checks.DrupalChecker
length_difference() (translate.lang.hi.hi class	method), 277
method), 369	long() (translate.filters.checks.GnomeChecker
<pre>length_difference() (translate.lang.hy.hy class     method), 370</pre>	method), 283

long() (translate.filters 289	checks.IOSChecker m	ethod),	makeheaderdi <i>late.stot</i>	.ct () rage.pocommon.pofile method),	( <i>trans</i> - 553
long() (translate.filters 294	checks.KdeChecker m	ethod),	makeheaderdi	.ct()	(trans- nethod),
long() (translate.filters.c	checks.L20nChecker m	ethod),	558	•	
300	ous obsolva Libus Office C	The alrea	makeheaderdi		(trans-
	ers.checks.LibreOfficeC	пескег		rage.poxliff.PoXliffFile method)	
method), 306	£ltll. Miil.	71 1	makeheaderdi		оо.ројне
	filters.checks.MinimalC	пескег	method		
<i>method</i> ), 312	Ch I I M 'II C	71 1	makeindex()	(translate.convert.csv2pe	o.csv2po
	.filters.checks.MozillaC	hecker	method		
<i>method</i> ), 317	1 1 0 000 0	71 1		(translate.storage.base.Translat	ionStore
	ers.checks.OpenOfficeC	hecker	method		
method), 323		~,	makeindex()	(translate.storage.catkeys.Cat	keysFile
	filters.checks.ReducedC	Checker	method		
method), 329			makeindex()	(translate.storage.csvl10	n.csvfile
-	ilters.checks.StandardC	Checker	method	, ,	
method), 335			<pre>makeindex()</pre>	(translate.storage.dtd.dtdfile 1	nethod),
long() (translate.filters.c	checks.TermChecker m	ethod),	427		
342			makeindex()(	(translate.storage.html.htmlfile 1	nethod),
lower() (translate	te.misc.multistring.mult	tistring	436		
method), 389			makeindex()	(translate.storage.html.POHTM	LParser
LRUCachingDict (class	in translate.misc.lru),	387	method		
lsep (in module translate.				(translate.storage.ical.icalfile 1	nethod).
	te.misc.multistring.mult	tistring	442	C	,,,
method), 389	8		makeindex()	(translate.storage.ini.inifile	nethod),
M				(translate.storage.jsonl10n.I18	NovtFilo
IVI					
		692			ivexii iie
main() (in module transla		682	method	), 452	
main() (in module transla main() (in module transla	ate.tools.pydiff), 698		<pre>method makeindex()</pre>	), 452 (translate.storage.jsonl10n.	
main() (in module transla main() (in module transla main() (in module transla	ate.tools.pydiff), 698 ate.tools.pypo2phppo),	698	<pre>method makeindex() method</pre>	), 452 (translate.storage.jsonl10n.	JsonFile
main() (in module transla main() (in module transla main() (in module transla make_postore_adder	ate.tools.pydiff), 698 ate.tools.pypo2phppo), () (in module		<pre>method makeindex()     method makeindex()</pre>	), 452 (translate.storage.jsonl10n. ), 457	IsonFile (trans-
main() (in module transla main() (in module transla main() (in module transla make_postore_adder	ate.tools.pydiff), 698 ate.tools.pypo2phppo),	698	<pre>method makeindex() method makeindex() late.sto</pre>	), 452 (translate.storage.jsonl10n.	IsonFile (trans-
main() (in module transla main() (in module transla main() (in module transla make_postore_adder late.storage.xml_a make_query()	ate.tools.pydiff), 698 ate.tools.pypo2phppo), () (in module extract.extract), 679	698 trans- (trans-	method makeindex() method makeindex() late.stor 459	), 452 (translate.storage.jsonl10n. ), 457	IsonFile (trans- nethod),
main() (in module transla main() (in module transla main() (in module transla make_postore_adder late.storage.xml_a make_query()	ate.tools.pydiff), 698 ate.tools.pypo2phppo), () (in module extract.extract), 679	698 trans- (trans-	method makeindex() method makeindex() late.stor 459	), 452 (translate.storage.jsonl10n. ), 457 rage.jsonl10n.JsonNestedFile	IsonFile (trans- nethod), (trans-
main() (in module transla main() (in module transla main() (in module transla make_postore_adder late.storage.xml_a make_query()	ate.tools.pydiff), 698 ate.tools.pypo2phppo), () (in module extract.extract), 679	698 trans- (trans-	method makeindex() method makeindex() late.stor 459 makeindex() late.stor	), 452 (translate.storage.jsonl10n. ), 457 rage.jsonl10n.JsonNestedFile r rage.jsonl10n.WebExtensionJso	IsonFile (trans- nethod), (trans-
main() (in module transla main() (in module transla main() (in module transla make_postore_adder late.storage.xml_a make_query() late.search.indexi method), 398 make_query()	ate.tools.pydiff), 698 ate.tools.pypo2phppo), () (in module extract.extract), 679 ing.CommonIndexer.Co	698 trans- (trans- mmonDa (trans-	method makeindex() method makeindex() late.stor 459 late.stor method	), 452 (translate.storage.jsonl10n. ), 457 rage.jsonl10n.JsonNestedFile rage.jsonl10n.WebExtensionJso ), 467	IsonFile (trans- nethod), (trans- nFile
main() (in module transla main() (in module transla main() (in module transla make_postore_adder late.storage.xml_a make_query() late.search.indexi method), 398 make_query()	ate.tools.pydiff), 698 ate.tools.pypo2phppo), () (in module extract.extract), 679 ing.CommonIndexer.Co	698 trans- (trans- mmonDa (trans-	method makeindex() method makeindex() late.stor 459 late.stor method	), 452 (translate.storage.jsonl10n. ), 457 rage.jsonl10n.JsonNestedFile rage.jsonl10n.WebExtensionJso ), 467	IsonFile (trans- nethod), (trans- nFile
main() (in module transla main() (in module transla main() (in module transla make_postore_adder late.storage.xml_a make_query() late.search.indexi method), 398 make_query()	ate.tools.pydiff), 698 ate.tools.pypo2phppo), () (in module extract.extract), 679 ing.CommonIndexer.Co	698 trans- (trans- mmonDa (trans-	method makeindex() method makeindex() late.stor 459 late.stor method	), 452 (translate.storage.jsonl10n. ), 457 rage.jsonl10n.JsonNestedFile r rage.jsonl10n.WebExtensionJso	IsonFile (trans- nethod), (trans- nFile
main() (in module transla main() (in module transla main() (in module transla make_postore_adder late.storage.xml_a make_query() late.search.indexi method), 398 make_query() late.search.indexi	ate.tools.pydiff), 698 ate.tools.pypo2phppo), () (in module extract.extract), 679 ing.CommonIndexer.Co	698 trans- (trans- mmonDa (trans- yLuceneL	method makeindex() method makeindex() late.ston 459 umbleeindex() late.ston method	), 452 (translate.storage.jsonl10n. ), 457 rage.jsonl10n.JsonNestedFile rage.jsonl10n.WebExtensionJso ), 467	JsonFile (trans- nethod), (trans- nFile nethod),
main() (in module transla main() (in module transla main() (in module transla make_postore_adder late.storage.xml_a make_query() late.search.indexi method), 398 make_query() late.search.indexi method), 401 make_query()	ate.tools.pydiff), 698 ate.tools.pypo2phppo), () (in module extract.extract), 679 ing.CommonIndexer.Co	698 trans- (trans- mmonDa (trans- yLuceneL (trans-	method makeindex() method makeindex() late.stor 459 late.stor method makeindex() late.stor method makeindex() 472 makeindex()	), 452 (translate.storage.jsonl10n. ), 457 rage.jsonl10n.JsonNestedFile rage.jsonl10n.WebExtensionJso ), 467 (translate.storage.lisa.LISAfile r	JsonFile (trans- nethod), (trans- nFile nethod),
main() (in module transla main() (in module transla main() (in module transla make_postore_adder late.storage.xml_a make_query() late.search.indexi method), 398 make_query() late.search.indexi method), 401 make_query() late.search.indexi	ate.tools.pydiff), 698 ate.tools.pypo2phppo), () (in module extract.extract), 679 ing.CommonIndexer.Co	698 trans- (trans- mmonDa (trans- yLuceneL (trans-	method makeindex() method makeindex() late.stor 459 late.stor method makeindex() late.stor method makeindex() 472 makeindex()	), 452 (translate.storage.jsonl10n. ), 457 rage.jsonl10n.JsonNestedFile rage.jsonl10n.WebExtensionJso ), 467 (translate.storage.lisa.LISAfile r	JsonFile (trans- nethod), (trans- nFile nethod),
main() (in module transla main() (in module transla main() (in module transla make_postore_adder late.storage.xml_a make_query() late.search.indexi method), 398 make_query() late.search.indexi method), 401 make_query() late.search.indexi method), 404	ate.tools.pydiff), 698 ate.tools.pypo2phppo), () (in module extract.extract), 679 ing.CommonIndexer.Co ing.PyLuceneIndexer.Py ing.XapianIndexer.Xapi	698 trans- (trans- mmonDa (trans- yLuceneL (trans- ianDatab	method makeindex() method makeindex() late.stor 459 method makeindex() late.stor method makeindex() 472 makeindex() ase 478 makeindex()	), 452 (translate.storage.jsonl10n.), 457  rage.jsonl10n.JsonNestedFile rage.jsonl10n.WebExtensionJso ), 467 (translate.storage.lisa.LISAfile ranslate.storage.mo.mofile ranslate.stora	IsonFile (trans- nethod), (trans- nFile nethod), nethod),
main() (in module translation main() (in module translation) (in make_query()	ate.tools.pydiff), 698 ate.tools.pypo2phppo), () (in module extract.extract), 679 ing.CommonIndexer.Co ing.PyLuceneIndexer.Py ing.XapianIndexer.Xapi	698 trans- (trans- mmonDa (trans- yLuceneL (trans- ianDatab	method makeindex() method makeindex() late.stor 459 method makeindex() late.stor method makeindex() 472 makeindex() ase 478 makeindex()	), 452 (translate.storage.jsonl10n.), 457 rage.jsonl10n.JsonNestedFile of rage.jsonl10n.WebExtensionJso ), 467 (translate.storage.lisa.LISAfile of (translate.storage.mo.mofile of	IsonFile (trans- nethod), (trans- nFile nethod), nethod), (trans-
main() (in module translation main() (in module translation) (in module transl	ate.tools.pydiff), 698 ate.tools.pypo2phppo), () (in module extract.extract), 679 ing.CommonIndexer.Co ing.PyLuceneIndexer.Py ing.XapianIndexer.Xapi te.storage.mo.mofile mater.	698 trans- (trans- mmonDa (trans- yLuceneL (trans- ianDatab ethod),	method makeindex() method makeindex() late.stor 459 makeindex() late.stor method makeindex() 472 makeindex() ase 478 makeindex() late.stor 483	), 452 (translate.storage.jsonl10n.), 457  rage.jsonl10n.JsonNestedFile rage.jsonl10n.WebExtensionJso ), 467 (translate.storage.lisa.LISAfile rage.mozilla_lang.LangStore rage.mozilla_lang.lang.lang.lang.lang.lang.lang.lang.	JsonFile (trans- nethod), (trans- nFile method), nethod), (trans- nethod),
main() (in module translation main() (in module translation) (in make_query()	ate.tools.pydiff), 698 ate.tools.pypo2phppo), () (in module extract.extract), 679 ing.CommonIndexer.Co ing.PyLuceneIndexer.Py ing.XapianIndexer.Xapi	698 trans- (trans- mmonDa (trans- yLuceneL (trans- ianDatab ethod),	method makeindex() method makeindex() late.stor 459 method late.stor method makeindex() late.stor method makeindex() 472 makeindex() ase 478 makeindex() late.stor 483 makeindex()	), 452 (translate.storage.jsonl10n.), 457  rage.jsonl10n.JsonNestedFile rage.jsonl10n.WebExtensionJso ), 467 (translate.storage.lisa.LISAfile rage.mozilla_lang.LangStore rage.mozilla_lang.lang.lang.langstore rage.mozilla_langstore rage.mozillangstore rage.mozillangstore rage.mozillangstore rage.mozilla	JsonFile (trans- nethod), (trans- nFile method), nethod), (trans- nethod),
main() (in module translation main() (in module translation) (in make_query()	ate.tools.pydiff), 698 ate.tools.pypo2phppo), () (in module extract.extract), 679 ing.CommonIndexer.Co ing.PyLuceneIndexer.Py ing.XapianIndexer.Xapi te.storage.mo.mofile module	698 trans- (trans- mmonDa (trans- yLuceneL (trans- ianDatab ethod), n.pofile	method makeindex() method makeindex() late.stor 459 makeindex() late.stor method makeindex() 472 makeindex() ase 478 makeindex() late.stor 483 makeindex() method	), 452 (translate.storage.jsonl10n.), 457  rage.jsonl10n.JsonNestedFile rage.jsonl10n.WebExtensionJso ), 467 (translate.storage.lisa.LISAfile rage.mozilla_lang.LangStore rage.mozilla_lang.lang.lang.langstore rage.mozilla_langstore rage.mozillangstore rage.mozillangstore rage.mozillangstore rage.mozilla	JsonFile (trans- nethod), (trans- nFile nethod), nethod), (trans- nethod),
main() (in module translation main() (in module translation) (in make_query()	ate.tools.pydiff), 698 ate.tools.pypo2phppo), () (in module extract.extract), 679 ing.CommonIndexer.Co ing.PyLuceneIndexer.Py ing.XapianIndexer.Xapi te.storage.mo.mofile module	698 trans- (trans- mmonDa (trans- yLuceneL (trans- ianDatab ethod), n.pofile	method makeindex() method makeindex() late.stor 459 method makeindex() late.stor method method makeindex() 472 makeindex() late.stor 483 makeindex() method makeindex() method makeindex()	), 452 (translate.storage.jsonl10n.), 457  rage.jsonl10n.JsonNestedFile rage.jsonl10n.WebExtensionJso ), 467 (translate.storage.lisa.LISAfile rage.mozilla_lang.LangStore rage.mozilla_lang.LangStore (translate.storage.omegat.Ome ), 488	JsonFile (trans- nethod), (trans- nFile nethod), (trans- nethod), (trans- nethod), egaTFile (trans-
main() (in module translation main() (in module translation) (in make_query()	ate.tools.pydiff), 698 ate.tools.pypo2phppo), () (in module extract.extract), 679 ing.CommonIndexer.Co ing.PyLuceneIndexer.Py ing.XapianIndexer.Xapi te.storage.mo.mofile module slate.storage.pocommonute.storage.poheader.po	698 trans- (trans- mmonDa (trans- yLuceneL (trans- ianDatab ethod), n.pofile	method makeindex() method makeindex() late.stor 459 method makeindex() late.stor method makeindex() ase 478 makeindex() late.stor 483 makeindex() method makeindex() late.stor late.stor	), 452 (translate.storage.jsonl10n.), 457  rage.jsonl10n.JsonNestedFile rage.jsonl10n.WebExtensionJso ), 467 (translate.storage.lisa.LISAfile rage.mozilla_lang.LangStore rage.mozilla_lang.LangStore (translate.storage.omegat.Ome ), 488	JsonFile (trans- nethod), (trans- nFile nethod), nethod), (trans- nethod),
main() (in module translation main() (in module translation) (in make_query()	ate.tools.pydiff), 698 ate.tools.pypo2phppo), () (in module extract.extract), 679 ing.CommonIndexer.Co ing.PyLuceneIndexer.Py ing.XapianIndexer.Xapi te.storage.mo.mofile module	698 trans- (trans- mmonDa (trans- yLuceneL (trans- ianDatab ethod), n.pofile	method makeindex() method makeindex() late.stor 459 method makeindex() late.stor method makeindex() 472 makeindex() late.stor 483 makeindex() method makeindex() late.stor 483 makeindex() method makeindex() late.stor 490	), 452 (translate.storage.jsonl10n.), 457  rage.jsonl10n.JsonNestedFile or rage.jsonl10n.WebExtensionJso ), 467 (translate.storage.lisa.LISAfile of translate.storage.mo.mofile of translate.storage.mo.mofile of translate.storage.omegat.Omega	JsonFile (trans- nethod), (trans- nethod), nethod), (trans- nethod), egaTFile (trans- nethod),
main() (in module translation main() (in module translation) (in make_query()	ate.tools.pydiff), 698 ate.tools.pypo2phppo), () (in module extract.extract), 679 ing.CommonIndexer.Co ing.PyLuceneIndexer.Py ing.XapianIndexer.Xapi te.storage.mo.mofile m slate.storage.pocommon ate.storage.poheader.po	698 trans- (trans- mmonDa (trans- yLuceneL ianDatab ethod), n.pofile cheader	method makeindex() method makeindex() late.stor 459 makeindex() late.stor method makeindex() 472 makeindex() late.stor 483 makeindex() method makeindex() late.stor 483 makeindex() method makeindex() makeindex() late.stor 490 makeindex()	), 452 (translate.storage.jsonl10n.), 457  rage.jsonl10n.JsonNestedFile rage.jsonl10n.WebExtensionJso ), 467 (translate.storage.lisa.LISAfile rage.mozilla_lang.LangStore rage.mozilla_lang.LangStore (translate.storage.omegat.Ome ), 488	JsonFile (trans- nethod), (trans- nethod), nethod), (trans- nethod), egaTFile (trans- nethod),
main() (in module translational) (in make_query()	ate.tools.pydiff), 698 ate.tools.pypo2phppo), () (in module extract.extract), 679 ing.CommonIndexer.Co ing.PyLuceneIndexer.Py ing.XapianIndexer.Xapi te.storage.mo.mofile m slate.storage.pocommon ate.storage.poheader.po	698 trans- (trans- mmonDa (trans- yLuceneL ianDatab ethod), n.pofile cheader	method makeindex() method makeindex() late.stor 459 method makeindex() late.stor method makeindex() ase 478 makeindex() late.stor 483 makeindex() method makeindex() late.stor 490 makeindex() 548	(translate.storage.jsonl10n.), 457  rage.jsonl10n.JsonNestedFile rage.jsonl10n.WebExtensionJson), 467 (translate.storage.lisa.LISAfile rage.mozilla_lang.LangStore rage.mozilla_lang.LangStore rage.mozilla_lang.LangStore rage.mozilla_lang.Comegat.Om	JsonFile (trans- nethod), (trans- nFile nethod), (trans- nethod), (trans- nethod), egaTFile (trans- nethod), nethod),
main() (in module translation main() (in module translation) (in make_query()	ate.tools.pydiff), 698 ate.tools.pypo2phppo), () (in module extract.extract), 679 ing.CommonIndexer.Co ing.PyLuceneIndexer.Py ing.XapianIndexer.Xapi ite.storage.mo.mofile module.storage.pocommon ate.storage.poheader.po ilate.storage.poxliff.PoX e.storage.pypo.pofile module.storage.pypo.pofile module.storage.pypo.pofile module.storage.pypo.pofile module.storage.pypo.pofile module.storage.pypo.pofile module.storage.pypo.pofile module.storage.pypo.pofile module.storage.pypo.pofile module.storage.pypo.pofile module.	698 trans- (trans- mmonDa (trans- yLuceneL (trans- ianDatab ethod), n.pofile cheader (diffFile ethod),	method makeindex() method makeindex() late.stor 459 method makeindex() late.stor method makeindex() ase 478 makeindex() late.stor 483 makeindex() method makeindex() late.stor 490 makeindex() 548 makeindex()	(translate.storage.jsonl10n.), 457  rage.jsonl10n.JsonNestedFile rage.jsonl10n.WebExtensionJso ), 467 (translate.storage.lisa.LISAfile rage.mozilla_lang.LangStore rage.mozilla_lang.LangStore rage.mozilla_lang.LangStore rage.omegat.OmegatFileTab rage.omegat.OmegaTFileTab rage.omegat.OmegaTFileTab rage.omegat.OmegaTFileTab rage.omegat.ome	JsonFile (trans- nethod), (trans- nFile nethod), (trans- nethod), (trans- nethod), egaTFile (trans- nethod), nethod),
main() (in module translational) (in make_query()	ate.tools.pydiff), 698 ate.tools.pypo2phppo), () (in module extract.extract), 679 ing.CommonIndexer.Co ing.PyLuceneIndexer.Py ing.XapianIndexer.Xapi te.storage.mo.mofile m slate.storage.pocommon ate.storage.poheader.po	698 trans- (trans- mmonDa (trans- yLuceneL (trans- ianDatab ethod), n.pofile cheader (diffFile ethod),	method makeindex() method makeindex() late.stor 459 method makeindex() late.stor method makeindex() ase 478 makeindex() late.stor 483 makeindex() method makeindex() late.stor 490 makeindex() 548	(translate.storage.jsonl10n.), 457  rage.jsonl10n.JsonNestedFile rage.jsonl10n.WebExtensionJso ), 467 (translate.storage.lisa.LISAfile rage.mozilla_lang.LangStore rage.mozilla_lang.LangStore rage.mozilla_lang.LangStore rage.omegat.OmegatFileTab rage.omegat.OmegaTFileTab rage.omegat.OmegaTFileTab rage.omegat.OmegaTFileTab rage.omegat.ome	JsonFile (trans- nethod), (trans- nFile nethod), nethod), (trans- nethod), egaTFile (trans- nethod), nethod),

method), 562	makeobsolete() (trans-
<pre>makeindex() (translate.storage.properties.javafile</pre>	late.storage.base.TranslationUnit method), 412
<pre>makeindex() (translate.storage.properties.javautf8file</pre>	makeobsolete() (trans- late.storage.catkeys.CatkeysUnit method),
<pre>makeindex() (translate.storage.properties.joomlafile</pre>	418 makeobsolete() (translate.storage.csvl10n.csvunit
<pre>makeindex() (translate.storage.properties.propfile</pre>	method), 423 makeobsolete() (translate.storage.dtd.dtdunit
<pre>makeindex() (translate.storage.properties.stringsfile</pre>	method), 430 makeobsolete() (translate.storage.html.htmlunit
makeindex() (trans-	method), 439
late.storage.properties.stringsutf8file method), 587	makeobsolete() (translate.storage.ical.icalunit method), 445
<pre>makeindex() (translate.storage.pypo.pofile method), 590</pre>	makeobsolete() (translate.storage.ini.iniunit method), 450
<pre>makeindex() (translate.storage.qm.qmfile method),</pre>	makeobsolete() (trans-
596 makeindex() (translate.storage.qph.QphFile method),	late.storage.jsonl10n.I18NextUnit method), 455
601	makeobsolete() (trans-
<pre>makeindex() (translate.storage.rc.rcfile method), 607 makeindex() (trans-</pre>	late.storage.jsonl10n.JsonNestedUnit method), 462
late. storage. subtitles. Adv SubStation Alpha File	makeobsolete() (trans-
method), 613	$late. storage. js on l10 n. Js on Unit \\ method),$
makeindex() (trans-	465
late.storage.subtitles.MicroDVDFile method), 615	makeobsolete() (trans- late.storage.jsonl10n.WebExtensionJsonUnit
<pre>makeindex() (translate.storage.subtitles.SubRipFile</pre>	method), 470
method), 616	makeobsolete() (translate.storage.lisa.LISAunit
makeindex() (trans- late.storage.subtitles.SubStationAlphaFile	method), 475 makeobsolete() (translate.storage.mo.mounit
<pre>method), 618 makeindex() (translate.storage.subtitles.SubtitleFile</pre>	method), 481 makeobsolete() (trans-
method), 620	$late. storage. mozilla\_lang. Lang Unit method),$
<pre>makeindex() (translate.storage.tbx.tbxfile method),</pre>	486
625 makeindex() (translate.storage.tiki.TikiStore method),	makeobsolete() (trans- late.storage.omegat.OmegaTUnit method),
630	493
makeindex() (translate.storage.tmx.tmxfile method), 635	makeobsolete() (translate.storage.php.phpunit method), 551
makeindex() (trans-	makeobsolete() (trans-
late.storage.trados.TradosTxtTmFile method), 644	late.storage.pocommon.pounit method), 556
makeindex() (translate.storage.ts2.tsfile method), 646	makeobsolete() (trans-
$\begin{tabular}{ll} {\it makeindex()} & ({\it translate.storage.txt.TxtFile} & {\it method}), \\ & 652 & \\ \end{tabular}$	late.storage.poxliff.PoXliffUnit method), 566
<pre>makeindex() (translate.storage.utx.UtxFile method),</pre>	makeobsolete() (trans-
657	late.storage.properties.propunit method),
makeindex() (trans-	583 (translate storage mme nounit
late.storage.wordfast.WordfastTMFile method), 667	makeobsolete() (translate.storage.pypo.pounit method), 593
makeindex() (translate.storage.xliff.xlifffile method), 674	makeobsolete() (translate.storage.qm.qmunit method), 599
makekey() (in module translate.storage.oo), 494	makeobsolete() (translate.storage.qph.QphUnit

<pre>method), 604 makeobsolete() (translate.storage.rc.rcunit</pre>	map () (translate.storage.placeables.interfaces.InvisiblePlaceable method), 520
method), 609	$\verb map ()  (translate.storage.placeables.interfaces.Masking Placeable$
makeobsolete() (trans-	method), 522
late.storage.subtitles.SubtitleUnit method), 623	map () (translate.storage.placeables.interfaces.ReplacementPlaceable method), 523
makeobsolete() (translate.storage.tbx.tbxunit method), 628	map() (translate.storage.placeables.interfaces.SubflowPlaceable method), 525
makeobsolete() (translate.storage.tiki.TikiUnit method), 633	map() (translate.storage.placeables.strelem.StringElem method), 527
makeobsolete() (translate.storage.tmx.tmxunit method), 638	map () (translate.storage.placeables.terminology.TerminologyPlaceable method), 529
makeobsolete() (trans-	map() (translate.storage.placeables.xliff.Bpt method),
•	
late.storage.trados.TradosUnit method),	531
642	map() (translate.storage.placeables.xliff.Bx method),
makeobsolete() (translate.storage.ts2.tsunit	536
method), 649	map() (translate.storage.placeables.xliff.Ept method),
<pre>makeobsolete() (translate.storage.txt.TxtUnit</pre>	533
method), 655	map() (translate.storage.placeables.xliff.Ex method),
<pre>makeobsolete() (translate.storage.utx.UtxUnit</pre>	538
method), 660	map() (translate.storage.placeables.xliff.G method), 539
makeobsolete() (trans-	map() (translate.storage.placeables.xliff.It method), 541
late.storage.wordfast.WordfastUnit method),	map() (translate.storage.placeables.xliff.Ph method),
671	544
makeobsolete() (translate.storage.xliff.xliffunit	map() (translate.storage.placeables.xliff.Sub method),
method), 677	543
ManPageOption (class in translate.misc.optrecurse), 390	map() (translate.storage.placeables.xliff.UnknownXML method), 546
<pre>map() (translate.storage.placeables.base.Bpt method),</pre>	map() (translate.storage.placeables.xliff.X method), 535
498	markapproved() (trans-
map() (translate.storage.placeables.base.Bx method),	late.storage.poxliff.PoXliffUnit method),
506	566
map() (translate.storage.placeables.base.Ept method),	markapproved() (translate.storage.xliff.xliffunit
500	method), 677
map() (translate.storage.placeables.base.Ex method),	markfuzzy() (translate.storage.base.TranslationUnit
508	method), 412
map() (translate.storage.placeables.base.G method), 505	markfuzzy() (translate.storage.catkeys.CatkeysUnit method), 418
map() (translate.storage.placeables.base.It method),	markfuzzy() (translate.storage.csvl10n.csvunit
503	method), 424
map() (translate.storage.placeables.base.Ph method), 502	markfuzzy() (translate.storage.dtd.dtdunit method), 430
map() (translate.storage.placeables.base.Sub method), 511	markfuzzy() (translate.storage.html.htmlunit method), 439
map() (translate.storage.placeables.base.X method), 510	markfuzzy() (translate.storage.ical.icalunit method), 445
map() (translate.storage.placeables.general.AltAttrPlacea	ablerkfuzzy() (translate.storage.ini.iniunit method),
	450 laweabkeuzzy () (translate.storage.jsonl10n.I18NextUnit
method), 515 map() (translate.storage.placeables.general.XMLTagPlaceables.general.xml	method), 455 cemblekfuzzy() (trans-
method), 516	late.storage.jsonl10n.JsonNestedUnit method),
map() (translate.storage.placeables.interfaces.BasePlace	
method), 518	markfuzzy() (translate.storage.jsonl10n.JsonUnit

method), 465	late.storage.base.TranslationUnit method),
markfuzzy() (trans-	412
late. storage. js on 110 n. Web Extension Js on Unit	markreviewneeded() (trans-
method), 470	late.storage.catkeys.CatkeysUnit method),
markfuzzy() (translate.storage.lisa.LISAunit	419
method), 475	markreviewneeded() (trans-
markfuzzy() (translate.storage.mo.mounit method),	late.storage.csvl10n.csvunit method), 424
481	<pre>markreviewneeded() (translate.storage.dtd.dtdunit</pre>
markfuzzy() (trans-	method), 430
late.storage.mozilla_lang.LangUnit method),	markreviewneeded() (trans-
486	late.storage.html.htmlunit method), 439
<pre>markfuzzy() (translate.storage.omegat.OmegaTUnit</pre>	markreviewneeded() (trans-
method), 493	late.storage.ical.icalunit method), 445
<pre>markfuzzy() (translate.storage.php.phpunit method),</pre>	<pre>markreviewneeded() (translate.storage.ini.iniunit</pre>
551	method), 450
markfuzzy() (translate.storage.pocommon.pounit	markreviewneeded() (trans-
method), 557	late.storage.jsonl10n.I18NextUnit method),
markfuzzy() (translate.storage.poxliff.PoXliffUnit	455
method), 566	markreviewneeded() (trans-
markfuzzy() (translate.storage.properties.propunit	late.storage.jsonl10n.JsonNestedUnit method),
method), 584	462
markfuzzy() (translate.storage.pypo.pounit method),	markreviewneeded() (trans-
593	late.storage.jsonl10n.JsonUnit method),
<pre>markfuzzy() (translate.storage.qm.qmunit method),</pre>	465
599	markreviewneeded() (trans-
markfuzzy() (translate.storage.qph.QphUnit	late.storage.jsonl10n.WebExtensionJsonUnit
method), 604	method), 470
markfuzzy() (translate.storage.rc.rcunit method), 609	markreviewneeded() (trans-
markfuzzy() (translate.storage.subtitles.SubtitleUnit	late.storage.lisa.LISAunit method), 475
method), 623	markreviewneeded() (translate.storage.mo.mounit
markfuzzy() (translate.storage.tbx.tbxunit method),	method), 481
628	markreviewneeded() (trans-
<pre>markfuzzy() (translate.storage.tiki.TikiUnit method),</pre>	late.storage.mozilla_lang.LangUnit method),
633	486
markfuzzy() (translate.storage.tmx.tmxunit method),	markreviewneeded() (trans-
638	late.storage.omegat.OmegaTUnit method),
markfuzzy() (translate.storage.trados.TradosUnit	493
method), 642	markreviewneeded() (trans-
markfuzzy() (translate.storage.ts2.tsunit method),	late.storage.php.phpunit method), 551
649	markreviewneeded() (trans-
<pre>markfuzzy() (translate.storage.txt.TxtUnit method),</pre>	late.storage.pocommon.pounit method),
655	557
markfuzzy() (translate.storage.utx.UtxUnit method),	markreviewneeded() (trans-
660	late.storage.poxliff.PoXliffUnit method),
markfuzzy() (trans-	566
late.storage.wordfast.WordfastUnit method),	markreviewneeded() (trans-
671	late.storage.properties.propunit method),
markfuzzy() (translate.storage.xliff.xliffunit method),	584
677	markreviewneeded() (trans-
MARKINGATTRS (translate.storage.html.htmlfile at-	late.storage.pypo.pounit method), 593
tribute), 435	markreviewneeded() (translate.storage.qm.qmunit
MARKINGTAGS (translate.storage.html.htmlfile at-	method), 599
tribute), 435	markreviewneeded() (trans-
markreviewneeded() (trans-	late.storage.qph.QphUnit method), 604
//	G II Z

	merge() (translate.storage.dtd.dtdunit method), 430
method), 609	merge() (translate.storage.html.htmlunit method), 439
markreviewneeded() (trans-	merge() (translate.storage.ical.icalunit method), 445
late.storage.subtitles.SubtitleUnit method),	merge() (translate.storage.ini.iniunit method), 450
623	merge() (translate.storage.jsonl10n.I18NextUnit
markreviewneeded() (translate.storage.tbx.tbxunit	method), 455
method), 628	merge() (translate.storage.jsonl10n.JsonNestedUnit
markreviewneeded() (trans-	method), 462
late.storage.tiki.TikiUnit method), 633	merge() (translate.storage.jsonl10n.JsonUnit method),
markreviewneeded() (trans-	465
late.storage.tmx.tmxunit method), 638	$\verb merge()  (translate.storage.jsonl 10n. WebExtension Json Unit$
markreviewneeded() (trans-	method), 470
late.storage.trados.TradosUnit method),	merge() (translate.storage.lisa.LISAunit method), 475
642	merge() (translate.storage.mo.mounit method), 481
<pre>markreviewneeded() (translate.storage.ts2.tsunit</pre>	merge() (translate.storage.mozilla_lang.LangUnit
method), 649	method), 486
<pre>markreviewneeded() (translate.storage.txt.TxtUnit</pre>	merge() (translate.storage.omegat.OmegaTUnit
method), 655	method), 493
markreviewneeded() (trans-	merge() (translate.storage.php.phpunit method), 551
late.storage.utx.UtxUnit method), 660	merge() (translate.storage.pocommon.pounit method),
markreviewneeded() (trans-	557
late.storage.wordfast.WordfastUnit method),	merge() (translate.storage.poxliff.PoXliffUnit method),
671	566
markreviewneeded() (trans-	merge() (translate.storage.properties.propunit
late.storage.xliff.xliffunit method), 677	method), 584
MaskingPlaceable (class in trans-	merge() (translate.storage.pypo.pounit method), 593
late.storage.placeables.interfaces), 520	merge() (translate.storage.qm.qmunit method), 599
match_entities() (trans-	merge() (translate.storage.qph.QphUnit method), 604
late.convert.accesskey.UnitMixer method),	merge() (translate.storage.rc.rcunit method), 609
232	merge() (translate.storage.subtitles.SubtitleUnit
match_fuzzy() (in module trans-	method), 623
late.tools.pretranslate), 697	merge() (translate.storage.tbx.tbxunit method), 628
<pre>match_header() (translate.storage.csvl10n.csvunit</pre>	merge() (translate.storage.tiki.TikiUnit method), 633
method), 424	merge() (translate.storage.tmx.tmxunit method), 638
match_source() (in module trans-	merge() (translate.storage.trados.TradosUnit method),
late.tools.pretranslate), 697	643
<pre>match_template_id() (in module trans-</pre>	merge() (translate.storage.ts2.tsunit method), 650
late.tools.pretranslate), 697	merge() (translate.storage.txt.TxtUnit method), 655
<pre>match_template_location() (in module trans-</pre>	merge() (translate.storage.utx.UtxUnit method), 660
late.tools.pretranslate), 697	merge() (translate.storage.wordfast.WordfastUnit
matcher (class in translate.search.match), 405	method), 671
matchers (translate.storage.placeables.terminology.Term	
attribute), 529	merge_on (translate.storage.base.TranslationStore at-
matches() (translate.search.match.matcher method),	tribute), 409
406	merge_on (translate.storage.catkeys.CatkeysFile
matches()(translate.search.match.terminologymatcher	attribute), 415
method), 407	merge_on (translate.storage.csvl10n.csvfile attribute),
memory() (in module translate.tools.pretranslate), 697	421
merge() (translate.storage.base.TranslationUnit	merge_on (translate.storage.dtd.dtdfile attribute), 427
method), 412	merge_on (translate.storage.html.htmlfile attribute),
merge() (translate.storage.catkeys.CatkeysUnit	436
method), 419	merge_on (translate.storage.html.POHTMLParser at- tribute), 433
merge() (translate.storage.csvl10n.csvunit method), 424	merge_on (translate.storage.ical.icalfile attribute), 442
コムコ	mer ge_on (nanmane.morage.teat.teatjue annionie), 442

merge_on (translate.storage.ini.inifile attribute), 447	merge_on (translate.storage.trados.TradosTxtTmFile
merge_on (translate.storage.jsonl10n.118NextFile at-	attribute), 644
tribute), 452	merge_on (translate.storage.ts2.tsfile attribute), 646
merge_on (translate.storage.jsonl10n.JsonFile at- tribute), 457	merge_on (translate.storage.txt.TxtFile attribute), 652 merge_on (translate.storage.utx.UtxFile attribute), 657
merge_on (translate.storage.jsonl10n.JsonNestedFile	merge_on (translate.storage.wordfast.WordfastTMFile
attribute), 459	attribute), 667
merge_on (translate.storage.jsonl10n.WebExtensionJson	
<pre>attribute), 467 merge_on (translate.storage.lisa.LISAfile attribute),</pre>	merge_store() (in module translate.convert.sub2po), 265
472	<pre>merge_store() (translate.convert.json2po.json2po</pre>
merge_on (translate.storage.mo.mofile attribute), 478	method), 244
$\verb merge_on  (translate.storage.mozilla_lang.LangStore $	<pre>merge_store() (translate.convert.rc2po.rc2po</pre>
attribute), 483	method), 265
merge_on (translate.storage.omegat.OmegaTFile at-	merge_store() (translate.convert.resx2po.resx2po
tribute), 488	method), 265
merge_on (translate.storage.omegat.OmegaTFileTab attribute), 490	merge_stores() (translate.convert.ical2po.ical2po method), 242
merge_on (translate.storage.php.phpfile attribute), 548	<pre>merge_stores() (translate.convert.ini2po.ini2po</pre>
$\verb merge_on  (translate.storage.pocommon.pofile   at-$	method), 243
tribute), 553	merge_stores() (trans-
merge_on (translate.storage.poxliff.PoXliffFile at-	late.convert.mozlang2po.lang2po method),
tribute), 562	244
merge_on (translate.storage.properties.javafile at-	merge_stores() (translate.convert.php2po.php2po
tribute), 576	method), 246
merge_on (translate.storage.properties.javautf8file attribute), 577	merge_stores() (translate.convert.po2ical.po2ical method), 247
merge_on (translate.storage.properties.joomlafile at-	<pre>merge_stores() (translate.convert.po2ini.po2ini</pre>
tribute), 579	method), 248
merge_on (translate.storage.properties.propfile at-	merge_stores() (translate.convert.po2txt.po2txt
tribute), 581	method), 258
merge_on (translate.storage.properties.stringsfile attribute), 585	merge_stores() (trans- late.convert.po2yaml.po2yaml method),
merge_on (translate.storage.properties.stringsutf8file	263 method),
attribute), 587	merge_stores() (translate.convert.txt2po.txt2po
merge_on (translate.storage.pypo.pofile attribute), 590	method), 267
merge_on (translate.storage.qm.qmfile attribute), 596	merge_stores() (trans-
merge_on (translate.storage.qph.QphFile attribute),	late.convert.yaml2po.yaml2po method),
601	268
merge_on (translate.storage.rc.rcfile attribute), 607	mergeheaders() (translate.storage.mo.mofile
$\verb merge_on  (translate.storage.subtitles.AdvSubStationAlphalles)                                      $	
attribute), 613	mergeheaders () (translate.storage.pocommon.pofile
merge_on (translate.storage.subtitles.MicroDVDFile	method), 554
attribute), 615	mergeheaders() (trans-
merge_on (translate.storage.subtitles.SubRipFile attribute), 617	late.storage.poheader.poheader method), 558
$\verb merge_on  (translate.storage.subtitles.SubStationAlphaFi $	$\emph{le}$ mergeheaders() ( $\emph{translate.storage.poxliff.PoXliffFile}$
attribute), 618	method), 562
merge_on (translate.storage.subtitles.SubtitleFile at-	mergeheaders() (translate.storage.pypo.pofile
tribute), 620	method), 590
merge_on (translate.storage.tbx.tbxfile attribute), 625	mergestore() (translate.convert.po2html.po2html
merge_on (translate.storage.tiki.TikiStore attribute), 630	<pre>method), 247 mergestore() (translate.convert.prop2po.prop2po</pre>
merge_on (translate.storage.tmx.tmxfile attribute), 635	method), 264
	, = 0 .

mergestores() (in module translate.tools.pomerge),	late.filters.helpers), 348
690	multistring (class in translate.misc.multistring), 387
MessageProgressBar (class in trans-	<pre>multistring_to_rich() (trans-</pre>
late.misc.progressbar), 394	late.storage.base.TranslationUnit method),
MicroDVDFile (class in translate.storage.subtitles),	412
614	<pre>multistring_to_rich() (trans-</pre>
Mimetypes (translate.storage.base.TranslationStore at-	late.storage.catkeys.CatkeysUnit method),
tribute), 408	419
MinimalChecker (class in translate.filters.checks),	<pre>multistring_to_rich() (trans-</pre>
309	late.storage.csvl10n.csvunit method), 424
miscpunc (translate.lang.common.Common attribute),	multistring_to_rich() (trans-
358	late.storage.dtd.dtdunit method), 430
mix_units() (translate.convert.accesskey.UnitMixer	
method), 232	late.storage.html.htmlunit method), 440
mkdir() (translate.convert.convert.ArchiveConvertOption	
method), 235	late.storage.ical.icalunit method), 445
mkdir()(translate.convert.convert.ConvertOptionParser	
method), 238	late.storage.ini.iniunit method), 450
mkdir()(translate.convert.po2moz.MozConvertOptionPa	
method), 250	late.storage.jsonl10n.118NextUnit method),
	456
mkdir() (translate.convert.po2tmx.TmxOptionParser	
method), 256	multistring_to_rich() (trans-
mkdir()(translate.convert.po2wordfast.WfOptionParser	
method), 260	462
mkdir() (translate.filters.pofilter.FilterOptionParser	
method), 349	late.storage.jsonl10n.JsonUnit method),
mkdir() (translate.misc.optrecurse.RecursiveOptionPars	
method), 391	multistring_to_rich() (trans-
mkdir() (translate.tools.poconflicts.ConflictOptionParse method), 684	r late.storage.jsonl10n.WebExtensionJsonUnit method), 470
mkdir() (translate.tools.pogrep.GrepOptionParser	<pre>multistring_to_rich() (trans-</pre>
method), 688	late.storage.lisa.LISAunit method), 475
mkdir()(translate.tools.porestructure.SplitOptionParser	The state of the s
method), 691	late.storage.mo.mounit method), 481
mkdir()(translate.tools.poterminology.TerminologyOpti	
method), 695	late.storage.mozilla_lang.LangUnit method),
ml (class in translate.lang.ml), 374	486
mofile (class in translate.storage.mo), 476	<pre>multistring_to_rich() (trans-</pre>
mounit (class in translate.storage.mo), 479	late.storage.omegat.OmegaTUnit method),
mounpack () (in module translate.storage.mo), 482	493
MozConvertOptionParser (class in trans-	multistring_to_rich() (trans-
late.convert.po2moz), 249	late.storage.php.phpunit method), 551
mozilla_pluralequation (trans-	multistring_to_rich() (trans-
late.lang.common.Common attribute), 358	late.storage.pocommon.pounit method),
MozillaChecker (class in translate.filters.checks),	557
315	
	multistring_to_rich() (trans-
mozillaescapemarginspaces() (in module translate.misc.quote), 395	late.storage.poxliff.PoXliffUnit method), 566
mr (class in translate.lang.mr), 375	multistring_to_rich() (trans-
msgidcomment (translate.storage.pypo.pounit at-	late.storage.properties.propunit method),
tribute), 593	584
<pre>multifilter() (in module translate.filters.helpers),</pre>	<pre>multistring_to_rich() (trans-</pre>
348	late.storage.pypo.pounit method), 593
multifiltertestmethod() (in module trans-	<pre>multistring_to_rich() (trans-</pre>

late.storage.qm.qmunit method), 599		317
<pre>multistring_to_rich()</pre>	(trans-	musttranslatewords() (trans-
late.storage.qph.QphUnit method), 60	)5	late.filters.checks.OpenOfficeChecker method),
multistring_to_rich()	(trans-	323
late.storage.rc.rcunit method), 610		musttranslatewords() (trans-
multistring_to_rich()	(trans-	late.filters.checks.ReducedChecker method),
late.storage.subtitles.SubtitleUnit	method),	329
623		musttranslatewords() (trans-
<pre>multistring_to_rich()</pre>	(trans-	late.filters.checks.StandardChecker method), 335
multistring_to_rich()	(trans-	musttranslatewords() (trans-
late.storage.tiki.TikiUnit method), 633	`	late.filters.checks.TermChecker method),
multistring_to_rich()	(trans-	342
late.storage.tmx.tmxunit method), 638		312
multistring_to_rich()	(trans-	N
	method),	
late.storage.trados.TradosUnit 643	meinoa),	Name (translate.storage.base.TranslationStore attribute), 408
multistring_to_rich()  late.storage.ts2.tsunit method), 650	(trans-	namespaced() (in module trans- late.misc.xml_helpers), 396
<pre>multistring_to_rich()     late.storage.txt.TxtUnit method), 655</pre>	(trans-	namespaced() (translate.storage.lisa.LISAfile method), 472
multistring_to_rich()	(trans-	namespaced() (translate.storage.lisa.LISAunit
late.storage.utx.UtxUnit method), 660		method), 475
<pre>multistring_to_rich()</pre>	(trans-	namespaced() (translate.storage.poxliff.PoXliffFile
late.storage.word fast.Word fast Unit	method),	method), 562
671		namespaced() (translate.storage.poxliff.PoXliffUnit
<pre>multistring_to_rich()</pre>	(trans-	method), 566
late.storage.xliff.xliffunit class 678	method),	namespaced() (translate.storage.qph.QphFile method), 601
musttranslatewords()	(trans-	namespaced() (translate.storage.qph.QphUnit
late.filters.checks.CCLicenseChecker	method),	method), 605
271		namespaced() (translate.storage.tbx.tbxfile method),
musttranslatewords()	(trans-	625
late.filters.checks.DrupalChecker 277	method),	namespaced() (translate.storage.tbx.tbxunit method), 628
musttranslatewords()	(trans-	namespaced() (translate.storage.tmx.tmxfile method),
late. filters. checks. Gnome Checker	method),	635
283		namespaced() (translate.storage.tmx.tmxunit
musttranslatewords()	(trans-	method), 639
late.filters.checks.IOSChecker method		namespaced() (translate.storage.ts2.tsfile method),
musttranslatewords()	(trans-	646
late.filters.checks.KdeChecker 294	method),	namespaced() (translate.storage.ts2.tsunit method), 650
musttranslatewords()	(trans-	namespaced() (translate.storage.xliff.xlifffile method),
late.filters.checks.L20nChecker	method),	674
300		namespaced() (translate.storage.xliff.xliffunit
musttranslatewords()	(trans-	method), 678
late.filters.checks.LibreOfficeChecker	•	native_distance() (in module trans-
306		late.search.lshtein), 405
musttranslatewords()	(trans-	ne (class in translate.lang.ne), 376
late.filters.checks.MinimalChecker 312	method),	newlines () (translate.filters.checks.CCLicenseChecker method), 271
musttranslatewords()	(trans-	
late.filters.checks.MozillaChecker	method),	newlines() (translate.filters.checks.DrupalChecker method). 277

newlines() (translate.filters.checks.GnomeChecker method), 283	late.filters.checks.LibreOfficeChecker method), 306
newlines() (translate.filters.checks.IOSChecker method), 289	notranslatewords() (trans- late.filters.checks.MinimalChecker method),
newlines() (translate.filters.checks.KdeChecker	312
method), 294	notranslatewords() (trans-
newlines() (translate.filters.checks.L20nChecker method), 300	late.filters.checks.MozillaChecker method), 318
newlines()(translate.filters.checks.LibreOfficeChecker	notranslatewords() (trans-
<pre>method), 306 newlines() (translate.filters.checks.MinimalChecker</pre>	late.filters.checks.OpenOfficeChecker method), 323
method), 312	notranslatewords() (trans-
newlines() (translate.filters.checks.MozillaChecker method), 318	late.filters.checks.ReducedChecker method), 329
newlines()(translate.filters.checks.OpenOfficeChecker	notranslatewords() (trans-
method), 323	late.filters.checks.StandardChecker method),
newlines() (translate.filters.checks.ReducedChecker	335
method), 329	notranslatewords() (trans-
newlines() (translate.filters.checks.StandardChecker method), 335	late.filters.checks.TermChecker method), 342
newlines() (translate.filters.checks.TermChecker method), 342	nplurals (translate.lang.common.Common attribute), 358
NoInitialStateError,672	$\verb nplurals   \textit{(translate.filters.checks.StandardUnitChecker} $
NoProgressBar (class in translate.misc.progressbar), 394	method), 339 numbers() (translate.filters.checks.CCLicenseChecker
normalize() (in module translate.lang.data), 360	method), 271
normalize_html() (in module trans- late.storage.html), 440	numbers() (translate.filters.checks.DrupalChecker method), 277
normalize_space() (in module trans- late.misc.xml_helpers), 396	numbers() (translate.filters.checks.GnomeChecker method), 283
normalize_xml_space() (in module trans- late.misc.xml_helpers), 396	numbers () (translate.filters.checks.IOSChecker method), 289
normalized_unicode() (in module trans-	numbers () (translate.filters.checks.KdeChecker
late.lang.data), 360	method), 295
normalizefilename() (in module trans- late.storage.oo), 494	numbers() (translate.filters.checks.L20nChecker method), 301
notranslatewords() (trans-	numbers () (translate.filters.checks.LibreOfficeChecker
late.filters.checks.CCLicenseChecker method), 271	method), 306 numbers () (translate.filters.checks.MinimalChecker
notranslatewords() (trans-	method), 312
late.filters.checks.DrupalChecker method), 277	numbers () (translate.filters.checks.MozillaChecker method), 318
notranslatewords() (trans-	numbers() (translate.filters.checks.OpenOfficeChecker
late.filters.checks.GnomeChecker method),	method), 323
283	numbers() (translate.filters.checks.ReducedChecker
notranslatewords() (trans-	method), 329
late.filters.checks.IOSChecker method), 289	numbers () (translate.filters.checks.StandardChecker
notranslatewords() (trans- late.filters.checks.KdeChecker method),	method), 335 numbers () (translate.filters.checks.TermChecker
295	method), 342
notranslatewords() (trans-	numbertranslate() (translate.lang.af.af class
late.filters.checks.L20nChecker method),	method), 353
300 notranslatewords() (trans-	numbertranslate() (translate.lang.am.am class method), 354
moeranistacewords () (mins-	manua), sor

<pre>numbertranslate()     method), 354</pre>	(translate.lang.ar.ar	class	numbertranslate() (translate.lang.te.te clamethod), 381	SS
<pre>numbertranslate()     method), 355</pre>	(translate.lang.bn.bn	class	numbertranslate() (translate.lang.th.th clamethod), 382	SS
<pre>numbertranslate()</pre>		(trans-	numbertranslate() (translate.lang.ug.ug cla	SS
late.lang.code_o	or.code_or class me	ethod),	method), 383	
356		(4	numbertranslate() (translate.lang.ur.ur cla	SS
numbertranslate()  late.lang.commo		(trans-	method), 384	
iate.tang.commo 358	on.Common class me	ethod),	numbertranslate() (translate.lang.vi.vi cla method), 385	33
numbertranslate()	(translate.lang.de.de	class	numbertranslate() (translate.lang.zh.zh cla	SS
method), 363	·		method), 385	
numbertranslate()	(translate.lang.el.el	class	numbertuple (translate.lang.common.Commo	on
method), 364			attribute), 358	
<pre>numbertranslate()</pre>	(translate.lang.es.es	class	numstart() (translate.lang.af.af class method), 353	
method), 365			numstart() (translate.lang.am.am class method), 35	
<pre>numbertranslate()</pre>	(translate.lang.fa.fa	class	numstart() (translate.lang.ar.ar class method), 354	
method), 366			numstart() (translate.lang.bn.bn class method), 355	
<pre>numbertranslate()</pre>	(translate.lang.fi.fi	class	<pre>numstart() (translate.lang.code_or.code_or cla</pre>	SS
method), 366		_	method), 356	
numbertranslate()	(translate.lang.fr.fr	class	numstart() (translate.lang.common.Common cla	SS
method), 367		,	method), 359	
numbertranslate()	(translate.lang.gu.gu	class	numstart () (translate.lang.de.de class method), 363	,
method), 368	/· 1 · 1 1 1	7	numstart () (translate.lang.el.el class method), 364	
numbertranslate()	(translate.lang.he.he	class	numstart () (translate.lang.es.es class method), 365	
<i>method</i> ), 369	(tuanalata lana hi hi	alaaa	numstart () (translate.lang.fa.fa class method), 366	
numbertranslate()	(translate.lang.hi.hi	class	numstart() (translate.lang.fi.fi class method), 366	
<pre>method), 370 numbertranslate()</pre>	(translate.lang.hy.hy	class	numstart() (translate.lang.fr.fr class method), 367 numstart() (translate.lang.gu.gu class method), 368	2
method), 370	(transtate.tang.ny.ny	ciuss	numstart() (translate.lang.he.he class method), 369	
numbertranslate()	(translate.lang.ja.ja	class	numstart () (translate.lang.hi.hi class method), 370	
<i>method</i> ), 371	(transtate.tang.ja.ja	Ciciss	numstart () (translate.lang.hy.hy class method), 370	)
numbertranslate()	(translate.lang.km.km	class	numstart () (translate.lang.ja.ja class method), 371	
method), 372	(		numstart() (translate.lang.km.km class method), 37	2
numbertranslate()	(translate.lang.kn.kn	class	numstart () (translate.lang.kn.kn class method), 373	
<i>method</i> ), 373	,		numstart() (translate.lang.ko.ko class method), 374	
<pre>numbertranslate()</pre>	(translate.lang.ko.ko	class	numstart() (translate.lang.ml.ml class method), 374	1
method), 374			numstart() (translate.lang.mr.mr class method), 375	5
<pre>numbertranslate()</pre>	(translate.lang.ml.ml	class	numstart() (translate.lang.ne.ne class method), 376	
method), 374			numstart() (translate.lang.pa.pa class method), 377	7
<pre>numbertranslate()</pre>	(translate.lang.mr.mr	class	numstart() (translate.lang.si.si class method), 378	
method), 375		_	numstart() (translate.lang.st.st class method), 379	
numbertranslate()	(translate.lang.ne.ne	class	numstart () (translate.lang.sv.sv class method), 380	
method), 376	(	7	numstart () (translate.lang.ta.ta class method), 381	
numbertranslate()	(translate.lang.pa.pa	class	numstart () (translate.lang.te.te class method), 382	
method), 377	(4	-1	numstart () (translate.lang.th.th class method), 382	,
numbertranslate()	(translate.lang.si.si	class	numstart() (translate.lang.ug.ug class method), 383	
method), 378	(translate lang et et	alass	numstart() (translate.lang.ur.ur class method), 384	
numbertranslate() <i>method</i> ), 379	(translate.lang.st.st	class	numstart() (translate.lang.vi.vi class method), 385 numstart() (translate.lang.zh.zh class method), 385	
numbertranslate()	(translate.lang.sv.sv	class	manus care () (mansume.umg.zn.zn ciuss memou), 363	
method), 380	(irunsiaic.iung.sv.sv	ciuss	0	
numbertranslate()	(translate.lang.ta.ta	class		) C
<i>method</i> ), 381	(		OMEGAT_FIELDNAMES (in module tran late.storage.omegat), 487	.s-
,,			inic.siorage.omegai), 40/	

OmegaTDialect (class 487	ss in translate.storage.omegat),	OpenOfficeChecker late.filters.checks)	`	in	trans-
OmegaTFile (class in	translate.storage.omegat), 487	openoutputfile()			(trans-
	ss in translate.storage.omegat),	late.convert.conve method), 235	rt.ArchiveConv	ertO	`
OmegaTUnit (class in	translate.storage.omegat), 491	openoutputfile()			(trans-
oofile (class in transle	ate.storage.oo), 495	late.convert.conve method), 238	ert.ConvertOptio	on <b>P</b> a	rser
	i translate.storage.oo), 495	openoutputfile()			(trans-
oounit (class in transle		late.convert.po2m	oz MozConvert	Onti	`
openarchive()	(trans-	method), 250	02.11/102.001110111	Opin	oni anser
	nvert.ArchiveConvertOptionParse				(trans-
method), 235		late.convert.po2tn	nx TmxOptionP	arsei	`
openarchive()	(trans-	method), 256	wii i nii o p nonii o		
	2tmx.TmxOptionParser	openoutputfile()			(trans-
method), 256		late.convert.po2w	ordfast WfOntio	onPa	`
openarchive()	(trans-	method), 260	orașasii rijopiie	,,,,,	
	2wordfast.WfOptionParser	openoutputfile()			(trans-
method), 260	2 wordjast. Njopitoli arser	late.filters.pofilter	:FilterOptionPa	rser	`
openinputfile()	(trans-	350	in the opinion of		,
	nvert.ArchiveConvertOptionParse				(trans-
method), 235	werns and converse prioris and	late.misc.optrecur	se RecursiveOr	otion	`
openinputfile()	(trans-	method), 391	senteem sive op		
	nvert.ConvertOptionParser	openoutputfile() (tr method), 496	ranslate.storage	.00.0	omultifile
openinputfile()	(trans-	openoutputfile()			(trans-
	2moz.MozConvertOptionParser	late.tools.poconfli method), 684	cts.ConflictOpt	ionP	`
openinputfile()	(trans-	openoutputfile()			(trans-
	2tmx.TmxOptionParser	late.tools.pogrep.	GrepOptionPar:	ser	method),
openinputfile()	(trans-	openoutputfile()			(trans-
	2wordfast.WfOptionParser	late.tools.porestru method), 691	ıcture.SplitOptio	onPa	`
openinputfile()	(trans-	openoutputfile()			(trans-
	ter.FilterOptionParser method),	late.tools.potermi. method), 695	nology.Termino	logy	`
openinputfile()	(trans-	opentemplatefile()			(trans-
	curse.RecursiveOptionParser	late.convert.conve method), 235	ert.ArchiveConv	ertO	ptionParser
openinputfile()  method), 495	(translate.storage.oo.oomultifile	opentemplatefile()  late.convert.conve	ert.ConvertOptic	onPa	(trans-
openinputfile()	(trans-	<i>method</i> ), 238	-		
late.tools.poco	nflicts.ConflictOptionParser	opentemplatefile()			(trans-
<pre>method), 684 openinputfile()</pre>	(trans-	late.convert.po2m method), 250	oz.MozConvert	Opti	onParser
	`				(trans-
688	ep.GrepOptionParser method),	opentemplatefile()  late.convert.po2tm	ny TmyOntionP	arcai	`
openinputfile()	(trans-	method), 256	іл. і нілОрионго	urser	
	structure.SplitOptionParser	opentemplatefile()			(trans-
method), 691	шисите. <i>эрш</i> ориот arser	late.convert.po2w	ordfast WfOnti.	on Da	`
openinputfile()	(trans-	method), 260	оғајазі. พ.10рис	тға	isei
	rminology.TerminologyOptionPar				(trans-
method), 695	пиноюду. тегниноюду Орнонг иг	late.filters.pofilter	:FilterOptionPa	ırser	`

350		method), 283
<pre>opentemplatefile()</pre>	(trans-	
late.misc.optrecurse.RecursiveOption	Parser	method), 289
method), 392		options() (translate.filters.checks.KdeChecker
<pre>opentemplatefile()</pre>	(trans-	method), 295
late.tools.poconflicts.ConflictOptionP method), 684	arser	options() (translate.filters.checks.L20nChecker method), 301
<pre>opentemplatefile()</pre>	(trans-	options()(translate.filters.checks.LibreOfficeChecker
late.tools.pogrep.GrepOptionParser	method),	method), 306
688		options() (translate.filters.checks.MinimalChecker
opentemplatefile()	(trans-	method), 312
method), 692		options () (translate.filters.checks.MozillaChecker method), 318
<pre>opentemplatefile()</pre>		options()(translate.filters.checks.OpenOfficeChecker
late.tools.poterminology.Terminology method), 695	OptionPars	per method), 324 options() (translate.filters.checks.ReducedChecker
opentempoutputfile()	(trans-	method), 329
late.convert.convert.ArchiveConvertO method), 235	ptionParse	roptions() (translate.filters.checks.StandardChecker method), 335
opentempoutputfile()		options() (translate.filters.checks.TermChecker
late.convert.convert.ConvertOptionPa	ırser	method), 342
method), 238	,	outputconflicts() (trans-
opentempoutputfile()	(trans-	late.tools.poconflicts.ConflictOptionParser
late.convert.po2moz.MozConvertOpti method), 250	onParser	<pre>method), 684 outputterminology() (trans-</pre>
opentempoutputfile()  late.convert.po2tmx.TmxOptionParse.	(trans-	late.tools.poterminology.TerminologyOptionParse method), 695
method), 256	(4	P
opentempoutputfile()	(trans-	
late.convert.po2wordfast.WfOptionPa method), 260	rser	pa (class in translate.lang.pa), 376
opentempoutputfile()	(trans-	parse() (in module translate.misc.ourdom), 393 parse() (in module trans-
late.filters.pofilter.FilterOptionParser		parse() (in module trans- late.storage.placeables.parse), 525
350	memen,	parse() (translate.storage.base.TranslationStore
opentempoutputfile()	(trans-	method), 409
late.misc.optrecurse.RecursiveOption method), 392	Parser	parse() (translate.storage.catkeys.CatkeysFile method), 416
opentempoutputfile()	(trans-	parse() (translate.storage.csvl10n.csvfile method), 421
late. to ols. poconflicts. Conflict Option Page 1999 and	arser	parse() (translate.storage.dtd.dtdfile method), 427
method), 684		parse() (translate.storage.dtd.dtdunit method), 430
opentempoutputfile()	(tuana	parse() (translate.storage.html.htmlfile method), 436
late tools nearen Gran Ontion Parser	(trans-	I == = = () (, , , , , , , , , , , , , , , ,
late.tools.pogrep.GrepOptionParser 688	method),	parse() (translate.storage.html.POHTMLParser method), 433
688 opentempoutputfile()	method), (trans-	parse() (translate.storage.html.POHTMLParser
688 opentempoutputfile() late.tools.porestructure.SplitOptionPo	method), (trans-	parse() (translate.storage.html.POHTMLParser method), 433
688 opentempoutputfile() late.tools.porestructure.SplitOptionPomethod), 692	method), (trans-	parse() (translate.storage.html.POHTMLParser method), 433 parse() (translate.storage.ical.icalfile method), 442 parse() (translate.storage.ini.inifile method), 447 parse() (translate.storage.jsonl10n.I18NextFile
688 opentempoutputfile()     late.tools.porestructure.SplitOptionPomethod), 692 opentempoutputfile()     late.tools.poterminology.Terminology	method), (trans- urser (trans-	parse() (translate.storage.html.POHTMLParser method), 433  parse() (translate.storage.ical.icalfile method), 442  parse() (translate.storage.ini.inifile method), 447  parse() (translate.storage.jsonl10n.I18NextFile method), 452  reparse() (translate.storage.jsonl10n.JsonFile method),
688 opentempoutputfile()     late.tools.porestructure.SplitOptionPomethod), 692 opentempoutputfile()     late.tools.poterminology.Terminology     method), 695	method), (trans- urser (trans- OptionPars	parse() (translate.storage.html.POHTMLParser method), 433  parse() (translate.storage.ical.icalfile method), 442  parse() (translate.storage.ini.inifile method), 447  parse() (translate.storage.jsonl10n.I18NextFile method), 452  reparse() (translate.storage.jsonl10n.JsonFile method), 457
688 opentempoutputfile()     late.tools.porestructure.SplitOptionPomethod), 692 opentempoutputfile()     late.tools.poterminology.Terminology	method), (trans- urser (trans- OptionPars	parse() (translate.storage.html.POHTMLParser method), 433  parse() (translate.storage.ical.icalfile method), 442  parse() (translate.storage.ini.inifile method), 447  parse() (translate.storage.jsonl10n.I18NextFile method), 452  Teparse() (translate.storage.jsonl10n.JsonFile method), 457  parse() (translate.storage.jsonl10n.JsonNestedFile
opentempoutputfile()	method), (trans- urser (trans- OptionPars seChecker	parse() (translate.storage.html.POHTMLParser method), 433  parse() (translate.storage.ical.icalfile method), 442  parse() (translate.storage.ini.inifile method), 447  parse() (translate.storage.jsonl10n.I18NextFile method), 452  reparse() (translate.storage.jsonl10n.JsonFile method), 457

<pre>parse() parse()</pre>	(translate.storage.mo.mofile method), 478 (translate.storage.mozilla_lang.LangStore	parse(	) (translate.storage.trados.TradosTxtTmF method), 644	ile
_	method), 483	parse(	) (translate.storage.ts2.tsfile method), 647	
parse()	(translate.storage.omegat.OmegaTFile		) (translate.storage.txt.TxtFile method), 652	
1	method), 488	parse(	$) \ (translate. storage. utx. UtxFile\ method), 657$	
parse()	(translate.storage.omegat.OmegaTFileTab method), 490	parse(	) (translate.storage.wordfast.WordfastTMF method), 667	ile
parse()	(translate.storage.oo.oofile method), 495	parse(	) (translate.storage.xliff.xlifffile method), 674	
parse()	(translate.storage.php.phpfile method), 548	parse_	args() (tran	ns-
	(translate.storage.placeables.general.AltAttrPlacelass method), 513	ceable	late.convert.convert.ArchiveConvertOptionPomethod), 235	arser
parse()	(translate.storage.placeables.general.XMLEntity	Blaceabl	eargs() (trai	ns-
	class method), 515		late.convert.convert.ConvertOptionParser	
parse()	(translate. storage. placeables. general. XMLTagParticles and the property of the property o	laceable	method), 239	
	class method), 516	parse_		ns-
	(translate.storage.placeables.strelem.StringElemclass method), 527	ı	late.convert.po2moz.MozConvertOptionParsomethod), 250	er
	(translate.storage.placeables.terminology.Termin	n <b>okog s</b> Æla		ns-
	class method), 529	-F 63	late.convert.po2tmx.TmxOptionParser	
	(translate.storage.placeables.xliff.UnknownXML		method), 256	
	class method), 546	parse_		ns-
	(translate.storage.pocommon.pofile method),	I	late.convert.po2wordfast.WfOptionParser	
	554		method), 261	
		parse_		ns-
_	562	Parse_	late.filters.pofilter.FilterOptionParser metho	
	(translate.storage.properties.javafile method),		350	α),
	576	parse_		ns-
parse()	(translate.storage.properties.javautf8file	parse_	late.misc.optrecurse.RecursiveOptionParser	15
	method), 577		method), 392	
parse()	(translate.storage.properties.joomlafile	parse_		nc_
	method), 579	parse_	late.tools.poconflicts.ConflictOptionParser	13-
	(translate.storage.properties.propfile method),		method), 684	
	581	parse_		ท ๓-
	(translate.storage.properties.stringsfile	parse_	late.tools.pogrep.GrepOptionParser methol	
parse()	method), 586		688	u),
parse()		parse_		10 C
	method), 587	parse_	late.tools.porestructure.SplitOptionParser	13-
	(translate.storage.pypo.pofile method), 590		method), 692	
	(translate.storage.qm.qmfile method), 596	narco		10 C
_	(translate.storage.qph.QphFile method), 601	parse_	late.tools.poterminology.TerminologyOptioni	
	(translate.storage.rc.rcfile method), 607		method), 695	larse
	(translate.storage.rc.rcjite method), 007 (translate.storage.subtitles.AdvSubStationAlpha.	Eila x a a		10 C
	method), 613	Tuarse_	late.storage.benchmark.TranslateBenchmark	
	(translate.storage.subtitles.MicroDVDFile		method), 413	ei
parse()	method), 615	2220	noinput() (trai	10 C
	(translate.storage.subtitles.SubRipFile	parse_	late.filters.pofilter.FilterOptionParser metho	
parse()			350	<i>u</i> ),
	method), 617  (translate storage subtitles SubStationAlphaFile	20200		10.0
	(translate.storage.subtitles.SubStationAlphaFile	harse_		
	method), 618  (translate storage subtitles Subtitle File		late.storage.benchmark.TranslateBenchmark	εI
parse()	(translate.storage.subtitles.SubtitleFile	20222	method), 413	<b>12</b> C
	method), 620 (translate storage the thefile method), 625	parse_		us-
	(translate.storage.tbx.tbxfile method), 625	Do	late.storage.xml_extract.misc), 680	
	(translate.storage.tiki.TikiStore method), 631		rror, 408	
parse()	(translate.storage.tmx.tmxfile method), 636	parsei	${\tt ile}$ () $(translate.storage.base.TranslationStorage)$	re

class method), 409	late.storage.properties.stringsutf8file class
parsefile() (translate.storage.catkeys.CatkeysFile	method), 587
class method), 416	parsefile() (translate.storage.pypo.pofile class
parsefile() (translate.storage.csvl10n.csvfile class method), 421	method), 590
**	parsefile() (translate.storage.qm.qmfile class method), 596
parsefile() (translate.storage.dtd.dtdfile class method), 427	
parsefile() (translate.storage.html.htmlfile class	parsefile() (translate.storage.qph.QphFile class method), 601
method), 436	parsefile() (translate.storage.rc.rcfile class
parsefile() (translate.storage.html.POHTMLParser	method), 607
class method), 434	parsefile() (trans-
parsefile() (translate.storage.ical.icalfile class	late.storage.subtitles.AdvSubStationAlphaFile
method), 442	class method), 613
parsefile() (translate.storage.ini.inifile class	
method), 447	late.storage.subtitles.MicroDVDFile class
	method), 615
parsefile() (translate.storage.jsonl10n.I18NextFile class method), 453	parsefile() (translate.storage.subtitles.SubRipFile
	class method), 617
parsefile() (translate.storage.jsonl10n.JsonFile class method), 457	parsefile() (trans-
parsefile() (trans-	late.storage.subtitles.SubStationAlphaFile
0 0	class method), 618
method), 459	parsefile() (translate.storage.subtitles.SubtitleFile
parsefile() (trans-	class method), 620
late.storage.jsonl10n.WebExtensionJsonFile class method), 467	parsefile() (translate.storage.tbx.tbxfile class method), 625
parsefile() (translate.storage.lisa.LISAfile class method), 472	method), 631
parsefile() (translate.storage.mo.mofile class method), 478	parsefile() (translate.storage.tmx.tmxfile class method), 636
parsefile() (trans-	parsefile() (trans-
late.storage.mozilla_lang.LangStore class	late.storage.trados.TradosTxtTmFile class
method), 483	method), 644
parsefile() (translate.storage.omegat.OmegaTFile	
class method), 489	method), 647
parsefile() (trans-	<pre>parsefile() (translate.storage.txt.TxtFile class</pre>
late.storage.omegat.OmegaTFileTab class	method), 652
method), 490	parsefile() (translate.storage.utx.UtxFile class
parsefile() (translate.storage.php.phpfile class	
method), 548	parsefile() (trans-
parsefile() (translate.storage.pocommon.pofile	late.storage.wordfast.WordfastTMFile class
class method), 554	method), 668
parsefile() (translate.storage.poxliff.PoXliffFile	<pre>parsefile() (translate.storage.xliff.xlifffile class</pre>
class method), 562	method), 674
parsefile() (translate.storage.properties.javafile	parseheader() (translate.storage.mo.mofile method),
class method), 576	478
parsefile() (translate.storage.properties.javautf8file	<pre>parseheader() (translate.storage.pocommon.pofile</pre>
class method), 578	method), 554
parsefile() (translate.storage.properties.joomlafile	parseheader() (trans-
class method), 579	late.storage.poheader.poheader method),
parsefile() (translate.storage.properties.propfile	559
class method), 581	parseheader() (translate.storage.poxliff.PoXliffFile
parsefile() (translate.storage.properties.stringsfile	method), 562
class method), 586	parseheader() (translate.storage.pypo.pofile
parsefile() (trans-	method), 590

parseheaderstring() (in module trans- late.storage.poheader), 558	<pre>class method), 562 parsestring() (translate.storage.properties.javafile</pre>
ParseState (class in trans-	class method), 576
late.storage.xml_extract.extract), 678	parsestring() (trans-
<pre>parseString() (in module translate.misc.ourdom),</pre>	late.storage.properties.javautf8file class
393	method), 578
parsestring() (trans-	1
late.storage.base.TranslationStore class method), 409	late.storage.properties.joomlafile class method), 579
parsestring() (trans-	<pre>parsestring() (translate.storage.properties.propfile</pre>
late.storage.catkeys.CatkeysFile class method),	class method), 581
416	parsestring() (trans-
<pre>parsestring() (translate.storage.csvl10n.csvfile</pre>	late.storage.properties.stringsfile class
class method), 421	method), 586
parsestring() (translate.storage.dtd.dtdfile class	parsestring() (trans-
method), 427	late.storage.properties.stringsutf8file class
parsestring() (translate.storage.html.htmlfile class	method), 587
method), 436	parsestring() (translate.storage.pypo.pofile class
parsestring() (trans-	method), 590
	parsestring() (translate.storage.qm.qmfile class
method), 434	method), 596
	parsestring() (translate.storage.qph.QphFile class
method), 442	method), 602
parsestring() (translate.storage.ini.inifile class	<i>"</i>
method), 447	method), 607
parsestring() (trans-	parsestring() (trans-
late.storage.jsonl10n.I18NextFile class	late.storage.subtitles.AdvSubStationAlphaFile
method), 453	class method), 613
parsestring() (translate.storage.jsonl10n.JsonFile	
class method), 457	late.storage.subtitles.MicroDVDFile class
parsestring() (trans-	method), 615
late.storage.jsonl10n.JsonNestedFile class	
method), 459	late.storage.subtitles.SubRipFile class method),
parsestring() (trans-	617
late.storage.jsonl10n.WebExtensionJsonFile	
class method), 467	late.storage.subtitles.SubStationAlphaFile
parsestring() (translate.storage.lisa.LISAfile class	class method), 618
method), 472	parsestring() (trans-
parsestring() (translate.storage.mo.mofile class	late.storage.subtitles.SubtitleFile class
method), 478	method), 620
parsestring() (trans-	parsestring() (translate.storage.tbx.tbxfile class
late.storage.mozilla_lang.LangStore class	method), 625
method), 483	<pre>parsestring() (translate.storage.tiki.TikiStore class</pre>
parsestring() (trans-	method), 631
late.storage.omegat.OmegaTFile class	<pre>parsestring() (translate.storage.tmx.tmxfile class</pre>
method), 489	method), 636
parsestring() (trans-	parsestring() (trans-
late.storage.omegat.OmegaTFileTab class	late.storage.trados.TradosTxtTmFile class
method), 490	method), 644
parsestring() (translate.storage.php.phpfile class	parsestring() (translate.storage.ts2.tsfile class
method), 548	method), 647
	parsestring() (translate.storage.txt.TxtFile class
class method), 554	method), 652
<pre>parsestring() (translate.storage.poxliff.PoXliffFile</pre>	parsestring() (translate.storage.utx.UtxFile class

late.convert.po2tmx. TmxOptionParser method), 239 parsestring() (translate.storage.stiff.xliffile class method), 638 parsestring() (translate.storage.stiff.xliffile class method), 639 prittion() (translate.misc.mullistring.mullistring method), 389 prittion() (translate.storage.placeables.base), 500 prittip() (translate.storage.placeables.base), 500 prittip() (translate.storage.php), 547 prittipencode () (in module translate.storage.php), 547 prittipencode () (in module translate.storage.php), 547 prittipencode () (in module translate.storage.php), 547 prittipencode () (in module translate.storage.phm), 547 prittipencode () (translate.storage.html.htmlfile method), 434 prittipencode () (translate.storage.html.htmlfile method), 434 prittipencode () (translate.storage.html.htmlfile method), 437 prittipencode () (translate.storage.html.htmlfile method), 436 prittipencode () (translate.storage.html.htmlfile method), 436 prittipencode () (translate.storage.html.htmlfile method), 239 prittipencode () (translate.storage.html.htmlf	method), 657	late.convert.convert.ConvertOptionParser
method), 668 parsestring () (translate.storage.xlif/xliffile class method), 674 partition () (translate.misc.multistring multistring method), 389 Ph (class in translate.storage.placeables.base), 500 Ph (class in translate.storage.placeables.base), 500 Phpedecode () (in module translate.storage.php), 547 Phippine (class in translate.storage.php), 549 Pi_escape () (translate.storage.html.himlfile method), 436  pi_escape () (translate.storage.html.POHTMLParser method), 434  pi_unescape () (translate.storage.html.POHTMLParser method), 437  pi_unescape () (translate.storage.html.POHTMLParser method), 437  pi_unescape () (translate.comvert.po2did), 246 po2html (class in translate.comvert.po2did), 246 po2html (class in translate.comvert.po2did), 246 po2html (class in translate.comvert.po2did), 247 po2ini () (in module translate.comvert.po2did), 247 po2ini () (in module translate.comvert.po2morf, muny), 263 po2lan (class in translate.comvert.po2morf, 247 po2ini () (in module translate.comvert.po2morf, 248 po2tiki (class in translate.comvert.po2morf, 248 po2tiki (class in translate.comvert.po2morf, 248 po2tiki (class in translate.storage.pomonmon, 552 pofile (class in translate.storage.pomon, 552 pofile (class in translate.storage.pomon, 552 pofile (class in translate.comvert.po2morf, 247 pozini () (in module translate.comvert.po2morf, 248 po2tiki (class in translate.comvert.po2morf, 248 po2tiki (class in translate.storage.pomorf, 258 po2tiki (class in trans		
method), 674 partition () (translate.misc.multistring multistring method), 389 Ph (class in translate.storage.placeables.base), 500 Ph (class in translate.storage.placeables.alif), 543 php2pe (class in translate.storage.placeables.alif), 543 php2pe (class in translate.storage.placeables.alif), 547 php1edecode () (in module translate.storage.php), 547 php1edecode () (in module translate.php1edecode), 248 pidecode () (in module translate.php1edecode), 248 pidecode () (in module translate.php1edecode), 248 pidecode () (in module translate.php1edecode), 248 php1edecode () (in module tr		
method), 674 partition () (translate.misc.multistring.multistring method), 389 Ph (class in translate.storage.placeables.abse), 500 Ph (class in translate.storage.placeables.abig), 543 php2po (class in translate.comert.php2po), 246 phpdecode () (in module translate.storage.php), 547 php1ile (class in translate.storage.php), 547 php1ile (class in translate.storage.php), 549 pi_escape () (translate.storage.phm), 549 pi_escape () (translate.storage.phm), 549 pi_escape () (translate.storage.phm), 549 pi_escape () (translate.storage.phm), 547 pi_escape () (translate.storage.phm), 547 pi_escape () (translate.storage.phm), 547 pi_escape () (translate.storage.phm), 547 pi_escape () (translate.storage.phml.POHTMLParser method), 344 pi_unescape () (translate.storage.html.htmlfile method), 344 pi_unescape () (translate.storage.html.pOHTMLParser method), 347 pi_unescape () (translate.comvert.po2tdd), 246 po2thml (class in translate.comvert.po2tdd), 246 po2thml (class in translate.comvert.po2tdd), 246 po2thml (class in translate.comvert.po2tdd), 247 po2iaci (class in translate.comvert.po2tml), 248 po2tiki (clas		
partition () (translate.misc.multistring method), 389 Ph (class in translate.storage.placeables.base), 500 Ph (class in translate.storage.placeables.base), 500 Ph (class in translate.storage.placeables.alif), 543 phppepe (class in translate.storage.php), 246 phpdecode () (in module translate.storage.php), 547 phppifile (class in translate.storage.php), 547 phppifile (class in translate.storage.php), 547 phpifile (class in translate.storage.php), 547 phpifile (class in translate.storage.php), 547 phpifile (class in translate.storage.php), 549 pi_escape () (translate.storage.php), 549 pi_escape () (translate.storage.php), 549 pi_escape () (translate.storage.html.POHTMLParser method), 434 pi_unescape () (translate.storage.html.htmlfile method), 437 pi_unescape () (translate.storage.html.htmlfile method), 437 pi_unescape () (translate.storage.html.POHTMLParser method), 434 pluralequation (translate.torage.phml.pohtmlParser method), 434 pluralequation (translate.comvert.po2ltml), 247 po2ital (class in translate.comvert.po2ltml), 247 po2ital (class in translate.comvert.po2min), 247 po2ital (class in translate.comvert.po2min), 247 po2ital (class in translate.comvert.po2mon), 559 po2yaml (class in translate.comvert.po2mon), 550 po2yaml (class in translate.comvert.po2mon), 550 po1tile (class in translate.comvert.po2mon), 550 po1tile (class in translate.po1tile (class i		
method), 389 Ph (class in translate.storage.placeables.base), 500 Ph (class in translate.storage.placeables.xiff), 543 Php2po (class in translate.convert.php2po), 246 Phpp1pole (class in translate.storage.php), 547 Phppnnode () (in module translate.storage.php), 547 Phppnnode () (in module translate.storage.php), 549 Php1pinit (class in translate.storage.php), 549 Php2pinit (class in translate.storage.php), 547 Php2pinit (class in translate.storage.php), 547 Php2pinit (class in translate.storage.php), 547 Php2pinit (class in translate.storage.php1pinit (translate.storage.php1pinit (translate.storage.php1pinit (translate.storage.php1pinit (translate.storage.php1pinit (translate.storage.php1pinit (translate.storage.php1pinit (translate.storage.php1pinit (translate.storage.php1pinit (transl		
Ph (class in translate.convert.php2po), 246   Php2po (class in translate.convert.php2po), 246   Php2po (class in translate.corvert.php2po), 246   Php6plo (class in translate.storage.php), 547   Phpencode () (in module translate.storage.php), 547   Php6plo (class in translate.storage.php), 547   Php6plo (class in translate.storage.php), 549   Php6plo (class in translate.convert.poHmILParser method), 434   Php6plo (class in translate.convert.poZidd), 246   Php6plo (class in translate.convert.poZi	-	
php2po (class in translate convert,php2po), 246 phpencode () (in module translate.storage.php), 547 phpencode () (in module translate.storage.php), 547 phpenfile (class in translate.storage.php), 549 phpunit (class in translate.storage.php), 549 phpunit (class in translate.storage.php), 549 pi_escape () (translate.storage.html.htmlfile method), 436 pi_escape () (translate.storage.html.POHTMLParser method), 434 pi_unescape () (translate.storage.html.htmlfile method), 437 pi_unescape () (translate.storage.html.htmlfile method), 434 pi_unescape () (translate.storage.phm), 207 pi_unescape () (translate.storage.phm), 207 pi_unescape () (translate.storage.phml.htmlfile method), 434 pi_unescape () (translate.storage.phml.htmlfile method), 436 pi_unescape () (translate.storage.phml.htmlfile method), 437 pi_unescape () (translate.storage.phml.htmlfile method), 434 pi_unescape () (translate.storage.phml.htmlfile method), 434 pi_unescape () (translate.storage.phml.htmlfile method), 434 pi_unescape () (translate.storage.pothill.htmlfile method), 434 pi_unescape () (translate.storage.pothill.htmlfile method), 436 po_lit (class in translate.convert.po_lit (translate.storage.potim), 247 po_lit (class in translate.convert.po_lit (translate.storage.potim), 247 po_lit (class in translate.convert.po_litml), 247 po_lit (class in translate.storage.potimny), 263 po_lit (class in translate.convert.po_litml), 247 po_lit (translate.storage.potimny), 263 po_lit (class in translate.storage.potimny), 263 po_lit (class in translate.storage.potimny), 263 po_lit	Ph (class in translate.storage.placeables.base), 500	potifyformat() (trans-
pbpelcode () (in module translate.storage.php), 547 phperiode () (in module translate.storage.php), 547 phpfile (class in translate.storage.php), 549 phpfile (class in translate.storage.php), 549 phperiode () (translate.storage.php), 549 phperiode () (translate.storage.phph, 547 phperiode () (translate.storage.phph, 547 phperiode () (translate.storage.phph, 547 phperiode () (translate.storage.phml.htmlfile method), 243 pride () (translate.storage.phml.htmlfile method), 244 pi_unescape () (translate.storage.phml.htmlfile method), 244 pi_unescape () (translate.storage.phml.htmlfile method), 246 po_2dtd (class in translate.convert.po_2dtd), 246 po_2dtd (class in translate.convert.po_2dtd), 246 po_2in () (in module translate.convert.po_2mo_thml), 247 po_2in () (in module translate.tribute), 593 po		
phypericle (class in translate.storage.php), 547 phyprile (class in translate.storage.php), 549 phyprile (class in translate.storage.php), 549 pi_escape() (translate.storage.html.htmlfile method), 436 pi_escape() (translate.storage.html.POHTMLParser method), 434 pi_unescape() (translate.storage.html.htmlfile method), 437 pi_unescape() (translate.storage.phyn), 547 po2ical (class in translate.convert.po2thml), 246 po2lthtl (class in translate.convert.po2thml), 247 po2ical (class in translate.convert.po2thml), 247 po2ical (class in translate.convert.po2ini), 247 po2ical (class in translate.convert.po2ini), 247 po2ini() (in module translate.convert.po2mozfumny), 263 po2lang (class in translate.convert.po2mozfumny), 263 po2lang (class in translate.convert.po2thil), 258 po2lang (class in translate.convert.po2thil), 258 po2txt (class in translate.convert.po2txi), 258 po2txt (class in translate.convert.po2txi), 258 po2txt (class in translate.convert.po2txi), 258 po1ile (class in translate.convert.po2txi), 258 po2txt (class in translate.convert.po2txi), 258 po2txt (class in translate.convert.po2txi), 258 po1ile (class in translate.convert.po2txi), 258 po1ile (class in translate.convert.po2txi), 258 po1ile (class in translate.convert.po2txi), 258 po2txt (class in translate.convert.po2txi), 258 po1ile (class in translate.convert.po2txi), 258 po1ile (class in translate.convert.po2txi), 258 po2txt (class in translate.convert.po2txi), 258 po1ile (class in translate.convert.po2txi), 258 po2txt (class in translate.convert.po2txi), 258 po1ile (class in translate.convert.po2txi), 258 po1ile (class in translate.convert.po2txi), 258 po2txt (class in translate.convert.po2txi), 258 po1ile (class in translate.convert.po2txi), 258 po1ile (		
phyphilic (class in translate.storage.php), 547 phypunit (class in translate.storage.phy), 549 pi_escape() (translate.storage.phml.htmlfile method), 436 pi_escape() (translate.storage.html.POHTMLParser method), 437 pi_unescape() (translate.storage.html.htmlfile method), 437 pi_unescape() (translate.storage.html.htmlfile method), 437 pi_unescape() (translate.ang.common.Common late.storage.phtml.POHTMLParser method), 434 pluralequation (translate.lang.common.Common attribute), 359 po2dtd (class in translate.convert.po2did), 246 po2html (class in translate.convert.po2liml), 247 po2inci (in module translate.convert.po2inl), 247 po2inci) (in module translate.convert.po2inly, 258 po2inl (class in translate.convert.po2inly, 25		
phynit (class in translate.storage.html.htmlfile method), and the method is translate.convert.po2thmll.htmlfile method in the method		
pi_escape() (translate.storage.html.htmlfile method), 434 pi_uescape() (translate.storage.html.htmlfile method), 434 pi_uescape() (translate.storage.html.htmlfile method), 437 pi_uescape() (translate.storage.html.htmlfile method), 436 pi_uescape() (translate.storage.html.htmlfile method), 437 pi_uescape() (translate.storage.html.htmlfile method), 438 pi_uescape() (translate.storage.html.htmlfile method), 439 pi_uescape() (translate.storage.html.htmlfile method), 436 pi_uescape() (translate.storage.pop.ml.html.htmlfile method), 436 pi_uescape() (translate.storage.pop.ml.html.htmlfile method), 436 pi_uescape() (translate.storage.pop.ml.html.htmlfile method), 436 pi_uescape() (translate.storage.pop.ml.htmlfile method), 436 pi_uescape() (translate.storage.pop.ater.polate.torage.pop.polater.polate.torage.pop.polater.polate.torage.pop.polater.polate.torage.pop.polater.polate.torage.pop.polater.polate.torage.polater.polate.		
late.tools.pretranslate, 697   pretranslate, 697   pretranslate, 597   pretranslate, 598   pretranslate, 597   pretranslate, 598   pretranslate,		
pi_escape() (translate.storage.html.POHTMLParser method), 434 pi_unescape() (translate.storage.html.htmlfile method), 437 pi_unescape() (translate.storage.html.POHTMLParser method), 438 pi_unescape() (translate.storage.html.POHTMLParser method), 437 pi_unescape() (translate.storage.html.POHTMLParser method), 438 pi_unescape() (translate.storage.html.POHTMLParser method), 496 popitem() (translate.storage.pyno.pounit attribute), 593 print_help() (translate.storage.pyno.pounit attribute), 593 print_help() (translate.convert.OnvertOptionParser method), 235 print_help() (translate.convert.ConvertOptionParser method), 235 print_help() (translate.convert.OnvertOptionParser method), 251 print_help() (translate.convert.polamaly, 247 polamale.convert.polamale.convert.polamaly, 248 polamale.convert.polam		
method), 434 pi_unescape() (translate.storage.html.htmlfile method), 437 pi_unescape() (translate.storage.html.htmlfile late.storage.html.POHTMLParser method), 437 pi_unescape() (translate.storage.html.POHTMLParser method), 235 pi_unescape() (translate.storage.pop.nounit atstribute), 397 pi_unescape() (translate.storage.pop.nounit atstribute), 397 pi_unescape() (translate.storage.pop.nounit atstribute), 398 pi_nth_belp() (translate.storage.pop.nounit atstribute), 399 print_help() (translate.convert.pop.nounit atstribute), 399 pop.nounit class in translate.convert.pop.nounit atstribute), 399 print_help() (translate.convert.ConvertOptionParser method), 235 pop.nounit class in translate.convert.pop.nounit atstribute), 399 print_help() (translate.convert.pop.nounit atstribute), 390 pop.nounit atstribute), 399 print_help() (translate.convert.pop.nounit atstribute), 390 print_help() (translate.convert.pop.nounit atstribute), 390 print_help() (translate.onvert.pop.nounit atstribute), 390 print_help() (translate.nounit.pop.nounit.pop.no		
method), 437   clate.tools.pretranslate), 697   pi_unescape() (translate.storage.html.POHTMLParser method), 434   pluralequation (translate.lang.common.Common attribute), 359   po2dtd (class in translate.convert.po2dtd), 246   po2inc() (in module translate.convert.prop2mozfunny), 263   po2ini (class in translate.convert.po2ini), 247   po2ini (class in translate.convert.po2ini), 248   po2itik (class in translate.convert.po2mozfunny), 263   po2ini (class in translate.convert.po2mozlang), 248   po2ini (c		
pi_unescape() (trans-	pi_unescape() (translate.storage.html.htmlfile	<pre>pretranslate_unit() (in module trans-</pre>
late.storage.html.POHTMLParser method), 434 434 434 pluralequation (translate.lang.common.Common attribute), 359 po2dtd (class in translate.convert.po2dtd), 246 po2lttml (class in translate.convert.po2ltml), 247 po2ioal (class in translate.convert.po2ical), 247 po2ioal (class in translate.convert.po2ical), 247 po2ioal (class in translate.convert.po2ion), 247 po2ioal (class in translate.convert.po2mozfumy), 263 po2ioni (in module translate.convert.po2mozfumy), 263 po2ioni (class in translate.convert.po2mozfumy), 263 po2ioni (class in translate.convert.po2mozfumy), 263 po2ioni (class in translate.convert.po2mozfumy), 263 po2ioni (in module translate.c	method), 437	late.tools.pretranslate), 697
Sprint_help()   (trans-   attribute), 359   po2dtd (class in translate.convert.po2dtd), 246     po2html (class in translate.convert.po2html), 247     po2inc() (in module trans-   late.convert.prop2mozfunny), 263     po2ini() (in module trans-   late.convert.po2mozfunny), 263     po2ini() (in module translate.convert.po2mozfunny), 264     po2ini() (class in translate.convert.po2mozfunny), 265     print_help() (translate.convert.po2mozfunny), 265     print_help() (translate.sorage.popamon), 352     point_help() (translate.misc.opterture.se.ecursiveOptionParser method), 392     point_	pi_unescape() (trans-	prev_source (translate.storage.pypo.pounit at-
pluralequation (translate.lang.common.Common attribute), 359 po2dtd (class in translate.convert.po2dtd), 246 po2html (class in translate.convert.po2tml), 247 po2inc() (in module translate.convert.po2ini), 247 po2ini (class in translate.convert.po2ini), 247 po2ini () (in module translate.convert.po2ini), 247 po2ini () (in module translate.convert.po2ini), 247 po2ini () (in module translate.convert.po2ini), 263 po2iti () (in module translate.convert.po2mozfunny), 263 po2iti () (in module translate.convert.po2mozfunny), 263 po2iti () (in module translate.convert.po2mozfunny), 263 po2ini (class in translate.convert.po2mozfunny), 263 po2ini () (in module translate.convert.po	late.storage.html.POHTMLParser method),	tribute), 593
method), 235  po2dtd(class in translate.convert.po2dtd), 246 po2html (class in translate.convert.po2ltml), 247 po2inc() (in module translate.convert.po2inl), 247 po2inc() (in module translate.convert.po2inl), 247 po2ini (class in translate.convert.po2inl), 247 po2ini (class in translate.convert.po2inl), 247 po2ini (in module translate.convert.po2mozfunny), 263 po2it (in module translate.convert.po2mozfunny), 263 po2it (in module translate.convert.po2mozfunny), 263 po2ini (class in translate.convert.po2mozfunny), 263 po2ini (class in translate.convert.po2mozfunny), 263 po2ini (class in translate.convert.po2mozfunny), 263 po2ini (in module translate.convert.po2mozfunny), 264 po2ini (in module translate.convert.po2mozfunny), 268 po2ini (in module translate.convert.po2mozfunny), 269 po2ini (in module translate.convert.po2mozfunny), 269 po2ini (in module translate.conver	434	<pre>print_help() (trans-</pre>
po2dtd (class in translate.convert.po2dtd), 246 po2ltml (class in translate.convert.po2ltml), 247 po2ical (class in translate.convert.po2ical), 247 po2inc () (in module translate.convert.po2ini), 247 po2ini (class in translate.convert.po2ini), 247 po2ini (class in translate.convert.po2ini), 247 po2ini () (in module translate.convert.po2ini), 247 po2ini () (in module translate.convert.po2mozfunny), 263 po2ini (class in translate.convert.po2mozfunny), 263 po2ini () (in module translate.convert.po2mozfunny), 263 po2ini () (lass in translate.storage.pophoader), 558 point help () (translate.storage.pophoader), 558 point help		
po2html (class in translate.convert.po2html), 247 po2ical (class in translate.convert.po2ical), 247 po2ical (class in translate.convert.po2ical), 247 po2inc () (in module trans- late.convert.prop2mozfunny), 263 po2ini (class in translate.convert.po2ini), 247 po2ini () (in module trans- late.convert.prop2mozfunny), 263 po2it () (in module translate.convert.prop2mozfunny), 263 po2itali (class in translate.convert.po2mozfunny), 263 po2itali (class in translate.convert.po2mozfunny), 263 po2itali (class in translate.convert.po2mozfunny), 264 po2tiki (class in translate.convert.po2mozlang), 248 po2tiki (class in translate.convert.po2tiki), 253 po2txt (class in translate.convert.po2tiki), 253 po2txt (class in translate.convert.po2tiki), 253 po2tyaml (class in translate.convert.po2yaml), 262 pofile (class in translate.storage.pocommon), 552 pofile (class in translate.storage.poheader), 558 poheader (class in translate.storage.poheader), 558 POHTMLParser (class in translate.storage.poheader), 496 popitem() (translate.misc.dictutils.cidict method), 386 popitem() (translate.storage.oo.unormalizechar method), 496 popitem() (translate.storage.potentorunorunorunorunorunorunorunorunorunorun		
po2ical (class in translate.convert.po2ical), 247 po2inc() (in module translate.convert.prop2mozfunny), 263 po2ini (class in translate.convert.po2ini), 247 po2ini () (in module translate.convert.po2ini), 247 po2ini () (in module translate.convert.pop2mozfunny), 263 po2it () (in module translate.convert.prop2mozfunny), 263 po2it () (in module translate.convert.po2mozfunny), 262 print_help() (translate.FilterOptionParser method), 390 print_help() (translate.sorage.pophoader), 558 late.tools.pogrep.GrepOptionParser method), 391 poit (in module translate.convert.po2mozfunny), 262 print_help() (translate.tools.pogrep.GrepOptionParser method), 392 poit (in module translate.sorage.poo.unormalizechar method), 386 poit (in module translate.convert.po2mozfunny), 263 poit (in module translate.convert.po2mozfunny),		
po2inc() (in module trans- late.convert.prop2mozfunny), 263 po2ini (class in translate.convert.po2ini), 247 po2ini () (in module translate.convert.po2ini), 247 po2ini () (in module translate.convert.po2mozfunny), 263 late.convert.prop2mozfunny), 263 po2it () (in module translate.convert.po2mozfunny), 263 po2ini () (in module translate.convert.po2mozfunny), 264 po2ini () (in module translate.convert.po2mozfunny), 265 po2ini () (in module translate.convert.po2mozfunny), 261 po2ini () (in module translate.convert.po2mozfunny), 263 po2ini () (in module translate.convert.po2mozfunny), 262 poini () (in module translate.convert.po2mozfunny), 262 print_help () (translate.siorage.popinonParser method), 386 poini () (in module translate.convert.po2mozfunny), 262 print_help () (translate.mic.opitint.popininate.popininate.popininate.popininate.popininate.popininate.popininate.popininate.popininate.popininate.popininate.popininate.popininate.popininate.popininate.po		
late.convert.prop2mozfunny), 263 po2ini (class in translate.convert.po2ini), 247 po2ini () (in module translate.convert.prop2mozfunny), 263 po2iti () (in module translate.convert.po2mozfunny), 263 po2iti () (class in translate.convert.po2mozfunny), 264 po2iti () (class in translate.convert.po2tixi), 258 po1iti () (class in translate.storage.popommon), 552 po1it () (class in translate.storage.popommon), 552 po1it () (translate.misc.dictutils.cidict method), 386 pop () (translate.misc.		
po2ini (class in translate.convert.po2ini), 247 po2ini () (in module translate.convert.prop2mozfunny), 263 po2it () (in module translate.convert.prop2mozfunny), 263 po2it () (in module translate.convert.po2mozfunny), 264 po2it (class in translate.convert.po2mozlang), 248 po2it (in module translate.convert.po2mozlang), 248 print_help() (translate.po2wordfast.WfOptionParser method), 350 print_help() (translate.po2wordfast.WfOptionParser method), 350 print_help() (translate.misc.optiter.FilterOptionParser method), 392 pop() (translate.misc.dictutils.cidict method), 386 popit (in module translate.convert.po2wordfast.WfOptionParser method), 392 popit (translate.storage.pocommon), 552 print_help() (translate.misc.optiter.po2wordfast.WfOptionParser method), 392 popit (translate.storage.popheader), 558 popit (translate.storage.poheader), 386 popit (translate.misc.dictutils.cidict method), 386 popit em() (translate.misc.dictutils.cidict method), 386 popit em() (translate.storage.oo.unormalizechar method), 496 popit em() (translate.storage.oo.unormalizechar method), 496 popit em() (translate.storage.oo.unormalizechar method), 496 popit em() (translate.storage.oo.unormalizechar method), 386 popit em() (translate.storage.oo.unormalizechar method), 386 popit em() (translate.storage.popit em() (translate.storage.popit em() (translate.storage.	1	
po2ini() (in module trans- late.convert.prop2mozfunny), 263 po2it() (in module translate.convert.prop2mozfunny), 263 po2lang (class in translate.convert.po2mozlang), 248 po2tiki (class in translate.convert.po2tiki), 253 po2txt (class in translate.convert.po2tiki), 258 po2yaml (class in translate.convert.po2yaml), 262 pofile (class in translate.storage.pocommon), 552 pofile (class in translate.storage.pocommon), 552 pofile (class in translate.storage.pocommon), 552 pofile (class in translate.storage.poheader), 558 POHTMLParser (class in translate.storage.poheader), 558 POHTMLParser (class in translate.storage.poheader), 558 pop() (translate.misc.dictutils.cidict method), 386 pop() (translate.storage.oo.unormalizechar method), 496 popitem() (translate.storage.oo.unormalizechar method), 496 popitem() (translate.storage.oo.unormalizechar method), 496 potifyformat() (translate.storage.oo.unormalizechar method), 392 potifyformat() (translate.storage.oo.unormalizechar method), 684 potifyformat() (translate.storage.oo.unormalizechar method), 496 potifyformat() (t		
late.convert.prop2mozfunny), 263 po2it () (in module translate.convert.prop2mozfunny), 263 po2lang (class in translate.convert.po2mozlang), 248 po2tiki (class in translate.convert.po2tiki), 253 po2txt (class in translate.convert.po2tiki), 253 po2tyt (class in translate.convert.po2tiki), 258 po5ile (class in translate.convert.po2yaml), 262 pofile (class in translate.storage.pocommon), 552 pofile (class in translate.storage.pocommon), 552 pofile (class in translate.storage.poponeader), 558 POHTMLParser (class in translate.storage.poheader), 558 POHTMLParser (class in translate.storage.on.unormalizechar method), 386 pop () (translate.misc.dictutils.cidict method), 386 pop () (translate.misc.dictutils.cidict method), 386 popitem () (translate.storage.oo.unormalizechar method), 496 potifyformat () (translate.storage.oo.unormalizechar method), 235  late.convert.po2tmx.TmxOptionParser method), 256 print_help () (translate.convert.po2wordfast.WfOptionParser method), 261 print_help () (translate.filterS.pofilter.FilterOptionParser method), 392 print_help () (translate.misc.optrecurse.RecursiveOptionParser method), 684 print_help () (translate.tools.poconflicts.ConflictOptionParser method), 684 print_help () (translate.tools.pogrep.GrepOptionParser method), 688		
po2it() (in module translate.convert.prop2mozfunny), 263 po2lang (class in translate.convert.po2mozlang), 248 po2tiki (class in translate.convert.po2tiki), 253 po2txt (class in translate.convert.po2txi), 258 po2yaml (class in translate.convert.po2yaml), 262 pofile (class in translate.storage.pocommon), 552 pofile (class in translate.storage.poheader), 558 POHTMLParser (class in translate.storage.poheader), 558 POHTMLParser (class in translate.storage.html), 432 pop() (translate.misc.dictutils.cidict method), 386 popitem() (translate.misc.dictutils.cidict method), 386 popitem() (translate.storage.oo.unormalizechar method), 496 potifyformat() (translate.storage.oo.unormalizechar method), 235  method), 256 print_help() (translate.filters.po2wordfast.WfOptionParser method), 261 print_help() (translate.filters.pofilter.FilterOptionParser method), 350 print_help() (translate.misc.optrecurse.RecursiveOptionParser method), 392 print_help() (translate.misc.optrecurse.RecursiveOptionParser method), 684 popitem() (translate.misc.dictutils.cidict method), 386 potifyformat() (translate.storage.oo.unormalizechar method), 496 potifyformat() (translate.storage.oo.unormalizechar method), 235  method), 256 print_help() (translate.misc.optrecurse.RecursiveOptionParser method), 392 print_help() (translate.tools.poconflicts.ConflictOptionParser method), 684 print_help() (translate.misc.optrecurse.RecursiveOptionParser method), 684 print_help() (translate.misc.optrecurse.RecursiveOptionParser method), 688 print_help() (translate.misc.optrecurse.RecursiveOptionParser method), 688 print_help() (translate.misc.optrecurse.RecursiveOptionParser method), 688 print_help() (translate.misc.optrecurse.RecursiveOptionParser method), 684 print_help() (translate.tools.poconflicts.ConflictOptionParser method), 688 print_help() (translate.misc.optrecurse.RecursiveOptionParser method), 688 print_help() (translate.misc.optrecurse.RecursiveOptionParser method), 688 print_help() (translate.storage.optrecurse.RecursiveOptionParser method), 688 print_he		
print_help() (trans-po2lang (class in translate.convert.po2mozlang), 248 po2tiki (class in translate.convert.po2tiki), 253 method), 261 po2txt (class in translate.convert.po2yaml), 262 pofile (class in translate.storage.pocommon), 552 pofile (class in translate.storage.pocommon), 552 pofile (class in translate.storage.popheader), 558 poHIMLParser (class in translate.storage.poheader), 558 POHTMLParser (class in translate.storage.poheader), 496 popitem() (translate.misc.dictutils.cidict method), 386 popitem() (translate.storage.oo.unormalizechar method), 496 potifyformat() (translate.storage.oo.unormalizechar method), 496 potifyformat() (translate.storage.oo.unormalizechar method), 235 print_help() (translate.storage.opheader), 558 late.convert.po2wordfast.WfOptionParser method), 261 print_help() (translate.filters.pofilter.FilterOptionParser method), 390 print_help() (translate.misc.optiteory.Filter.PoptionParser method), 392 print_help() (translate.misc.optiteory.Filter.PoptionParser method), 392 print_help() (translate.storage.oo.unormalizechar method), 684 print_help() (translate.misc.optiteory.Filter.PoptionParser method), 392 print_help() (translate.misc.optiteory.Filter.PoptionParser method), 392 print_help() (translate.misc.optiteory.Filter.PoptionParser method), 392 print_help() (translate.misc.optiteory.Filter.PoptionParser method), 684 print_help() (translate.misc.optiteory.Filter.PoptionParser method), 684 print_help() (translate.misc.optiteory.Filter.PoptionParser method), 684 print_help() (translate.misc.optiteory.Filter.PoptionParser method), 688 print_help() (translate.storage.oo.unormalizechar method), 688 print_help() (translate.storage.oo.unormalizechar method), 386 print_help() (tran		
po2lang (class in translate.convert.po2mozlang), 248 po2tiki (class in translate.convert.po2tiki), 253 po2txt (class in translate.convert.po2tiki), 258 po2txt (class in translate.convert.po2txt), 258 pofile (class in translate.convert.po2yaml), 262 pofile (class in translate.storage.pocommon), 552 pofile (class in translate.storage.pocommon), 552 pofile (class in translate.storage.popo), 588 poHTMLParser (class in translate.storage.html), 432 pop () (translate.misc.dictutils.cidict method), 386 popitem () (translate.storage.oo.unormalizechar method), 496 popitem () (translate.storage.oo.unormalizechar method), 496 potifyformat () (translate.storage.oo.unormalizechar method), 496 potifyformat () (translate.storage.oo.unormalizechar method), 235  late.convert.po2wordfast.WfOptionParser method), 261 print_help() (trans-late.filters.pofilter.FilterOptionParser method), 390 print_help() (translate.misc.optrecurse.RecursiveOptionParser method), 684 print_help() (translate.tools.poconflicts.ConflictOptionParser method), 688 potifyformat () (translate.storage.oo.unormalizechar method), 496 potifyformat () (translate.storage.oo.unormalizechar method), 496 potifyformat () (translate.storage.oo.unormalizechar method), 496 potifyformat () (translate.storage.oo.unormalizechar method), 688 potifyformat () (translate.storage.oo.unormalizechar method), 688 potifyformat () (translate.storage.oo.unormalizechar method), 688 potifyformat () (translate.storage.oo.unormalizechar method), 692		
po2tiki (class in translate.convert.po2tiki), 253 po2txt (class in translate.convert.po2txt), 258 po2yaml (class in translate.convert.po2yaml), 262 pofile (class in translate.storage.pocommon), 552 pofile (class in translate.storage.pocommon), 552 pofile (class in translate.storage.popo), 588 poheader (class in translate.storage.poheader), 558 POHTMLParser (class in translate.storage.poheader), 558 POHTMLParser (class in translate.storage.html), 432 pop () (translate.misc.dictutils.cidict method), 386 pop () (translate.storage.oo.unormalizechar method), 496 popitem () (translate.misc.dictutils.cidict method), 386 potifyformat () (translate.storage.oo.unormalizechar method), 496 potifyformat () (translate.storage.oo.u	200	
po2txt (class in translate.convert.po2txt), 258 po2yaml (class in translate.convert.po2yaml), 262 pofile (class in translate.storage.pocommon), 552 pofile (class in translate.storage.popo), 588 poheader (class in translate.storage.poheader), 558 POHTMLParser (class in translate.storage.html), 432 pop () (translate.misc.dictutils.cidict method), 386 pop () (translate.storage.oo.unormalizechar method), 496 pop item () (translate.misc.dictutils.cidict method), 386 potifyformat () (translate.storage.oo.unormalizechar method), 496 potifyformat () (translate.storage.oo.unormalizechar method),		
po2yaml (class in translate.convert.po2yaml), 262 pofile (class in translate.storage.pocommon), 552 pofile (class in translate.storage.popo), 588 poheader (class in translate.storage.poheader), 558 POHTMLParser (class in translate.storage.html), 432 pop() (translate.misc.dictutils.cidict method), 386 pop() (translate.storage.oo.unormalizechar method), 496 popitem() (translate.misc.dictutils.cidict method), 386 potifyformat() (translate.storage.oo.unormalizechar method), 496 potifyf		
pofile (class in translate.storage.pocommon), 552 pofile (class in translate.storage.pypo), 588 poheader (class in translate.storage.poheader), 558 POHTMLParser (class in translate.storage.html), 432 pop () (translate.misc.dictutils.cidict method), 386 pop () (translate.storage.oo.unormalizechar method), 496 popitem () (translate.misc.dictutils.cidict method), 386 potifyformat () (translate.storage.oo.unormalizechar method), 496 potifyformat () (translate.storage.oo.unormalizechar method), 688 potifyformat () (translate.storage.oo.unormalizechar method), 688 potifyformat () (translate.storage.oo.unormalizechar method), 692		
pofile (class in translate.storage.pypo), 588 print_help() (transpoheader (class in translate.storage.poheader), 558		
poheader (class in translate.storage.poheader), 558 POHTMLParser (class in translate.storage.html), 432 pop () (translate.misc.dictutils.cidict method), 386 pop () (translate.storage.oo.unormalizechar method), 496 popitem () (translate.misc.dictutils.cidict method), 386 popitem () (translate.storage.oo.unormalizechar method), 496 popitem () (translate.storage.oo.unormalizechar method), 496 potifyformat () (translate.storage.oo.unormalizechar method), 688		
POHTMLParser (class in translate.storage.html), 432 method), 392  pop() (translate.misc.dictutils.cidict method), 386 print_help() (translate.storage.oo.unormalizechar method), 496 popitem() (translate.misc.dictutils.cidict method), 386 potifyformat() (translate.storage.oo.unormalizechar method), 496 potifyformat() (translate.convert.ArchiveConvertOptionParser method), 692  method), 392  print_help() (translate.storage.oo.flicts.ConflictOptionParser method), 684  print_help() (translate.tools.poconflicts.ConflictOptionParser method), 688  print_help() (translate.tools.pogrep.GrepOptionParser method), 688  potifyformat() (translate.storage.oo.unormalizechar method), 688  potifyformat() (translate.convert.ArchiveConvertOptionParser method), 692		
<pre>pop() (translate.misc.dictutils.cidict method), 386</pre>		
pop() (translate.storage.oo.unormalizechar method), 496 method), 684  popitem() (translate.misc.dictutils.cidict method), 386 print_help() (translate.storage.oo.unormalizechar method), 496 (688  potifyformat() (trans- print_help() (translate.convert.convert.ArchiveConvertOptionParser method), 692  method), 692		
496		
popitem() (translate.storage.oo.unormalizechar late.tools.pogrep.GrepOptionParser method), 496 688  potifyformat() (trans- print_help() (trans- late.convert.convert.ArchiveConvertOptionParser method), 235 late.tools.porestructure.SplitOptionParser method), 692		
popitem() (translate.storage.oo.unormalizechar late.tools.pogrep.GrepOptionParser method), 496 688  potifyformat() (trans- print_help() (trans- late.convert.convert.ArchiveConvertOptionParser method), 235 late.tools.porestructure.SplitOptionParser method), 692	<pre>popitem() (translate.misc.dictutils.cidict method), 386</pre>	<pre>print_help() (trans-</pre>
potifyformat() (trans- print_help() (trans- late.convert.convert.ArchiveConvertOptionParser method), 235 late.tools.porestructure.SplitOptionParser method), 692		
late.convert.convert.ArchiveConvertOptionParser late.tools.porestructure.SplitOptionParser method), 235 late.tools.porestructure.SplitOptionParser method), 692		
method), 235 method), 692		

	late.tools.poterminology.TerminologyOptionPars	er	late.storage.placeables.general.AltAttrP	laceable
	method), 695		method), 513	7.
print_		print_		(trans-
	late.convert.convert.ArchiveConvertOptionParsemethod), 235	r	late.storage.placeables.general.XMLEnt method), 515	ityPlaceable
print		print_		(trans-
P	late.convert.convert.ConvertOptionParser	P	late.storage.placeables.general.XMLTag	`
	method), 239		method), 517	1 iaccasic
print	manpage() (trans-	print_t		(trans-
	late.convert.po2moz.MozConvertOptionParser	_	late.storage.placeables.interfaces.BaseF	Placeable
	method), 251		method), 518	
print		print_t		(trans-
	late.convert.po2tmx.TmxOptionParser	_	late.storage.placeables.interfaces.Invisit	`
	method), 256		method), 520	
print.	manpage() (trans-	print_t		(trans-
r	late.convert.po2wordfast.WfOptionParser	1	late.storage.placeables.interfaces.Maski	`
	method), 261		method), 522	
print_	manpage() (trans-	print_t		(trans-
_	late.filters.pofilter.FilterOptionParser method),	_	late.storage.placeables.interfaces.Repla	cementPlaceable
	350		method), 523	
print_	manpage() (trans-	print_t		(trans-
_	late.misc.optrecurse.RecursiveOptionParser	_	late.storage.placeables.interfaces.Subflo	wPlaceable
	method), 392		method), 525	
print_	manpage() (trans-	print_t		(trans-
_	late.tools.poconflicts.ConflictOptionParser	_	late.storage.placeables.strelem.StringEl	em
	method), 684		method), 528	
print_	manpage() (trans-	print_t		(trans-
	late.tools.pogrep.GrepOptionParser method),		late.storage.placeables.terminology.Terr	ninologyPlaceable
	688		method), 530	
print_	manpage() (trans-	print_t	tree() (translate.storage.placeables.x	cliff.Bpt
	late.tools.porestructure.SplitOptionParser		method), 531	
	method), 692	print_t	tree() (translate.storage.placeables.	xliff.Bx
print_	manpage() (trans-		method), 536	
	late. tools. poterminology. Terminology Option Pars	eprint_t	tree() (translate.storage.placeables.x	cliff.Ept
	method), 695		method), 533	
print_	tree() (translate.storage.placeables.base.Bpt	print_	tree() (translate.storage.placeables.	xliff.Ex
	method), 498		method), 538	
print_	tree() (translate.storage.placeables.base.Bx			s.xliff.G
	method), 506		method), 539	
print_	tree() (translate.storage.placeables.base.Ept	print_t		s.xliff.It
	method), 500		method), 541	
print_	tree() (translate.storage.placeables.base.Ex	print_	tree() (translate.storage.placeables.	xliff.Ph
	method), 508		method), 544	
print_	tree() (translate.storage.placeables.base.G	print_t	tree() (translate.storage.placeables.x	:liff.Sub
	method), 505		method), 543	
print_	tree() (translate.storage.placeables.base.It	print_t		(trans-
	method), 503		late.storage.placeables.xliff.UnknownXN	AL
print_	tree() (translate.storage.placeables.base.Ph		method), 546	11.00 **
	method), 502	print_	tree() (translate.storage.placeable	s.xliff.X
print_	tree() (translate.storage.placeables.base.Sub		method), 535	
	method), 511	print_		(trans-
print_	tree() (translate.storage.placeables.base.X		late.convert.convert.ArchiveConvertOpt	onParser
	method), 510		method), 235	(4
print_	tree() (trans-	print_	usage()	(trans-

late.convert.convert.ConvertOptionParser method), 239	late.tools.pogrep.GrepOptionParser method), 688
print_usage() (tran	s- print_version() (trans-
late.convert.po2moz.MozConvertOptionParse	
method), 251	method), 692
print_usage() (tran.	
late.convert.po2tmx.TmxOptionParser	late.tools.poterminology.TerminologyOptionParser
method), 256	method), 696
print_usage() (trans	method), 272
late.convert.po2wordfast.WfOptionParser	
method), 261	printf() (translate.filters.checks.DrupalChecker
print_usage() (tran.	<i>"</i>
late.filters.pofilter.FilterOptionParser method	
350	method), 283
print_usage() (trans	i v
late.misc.optrecurse.RecursiveOptionParser	method), 289
method), 392	printf() (translate.filters.checks.KdeChecker
print_usage() (trans	
late.tools.poconflicts.ConflictOptionParser	printf() (translate.filters.checks.L20nChecker
method), 685	method), 301
<pre>print_usage() (trans</pre>	1
late.tools.pogrep.GrepOptionParser method	), <i>method</i> ), 306
688	<pre>printf() (translate.filters.checks.MinimalChecker</pre>
<pre>print_usage()</pre>	s- <i>method</i> ), 312
late.tools.porestructure.SplitOptionParser	<pre>printf() (translate.filters.checks.MozillaChecker</pre>
method), 692	method), 318
<pre>print_usage() (tran.</pre>	s- printf() (translate.filters.checks.OpenOfficeChecker
late.tools.poterminology.TerminologyOptionPo	
method), 695	printf() (translate.filters.checks.ReducedChecker
<pre>print_version() (tran.</pre>	
late.convert.convert.ArchiveConvertOptionPar	
method), 235	method), 335
print_version() (tran.	
late.convert.convert.ConvertOptionParser	method), 342
method), 239	process_translatable() (in module trans-
print_version() (trans	<del>-</del>
late.convert.po2moz.MozConvertOptionParse	
method), 251	late.convert.convert.ArchiveConvertOptionParser
_ ` ` ` ` ` ` ` ` ` ` ` ` ` ` ` ` ` ` `	
late.convert.po2tmx.TmxOptionParser	processfile() (trans-
method), 256	late.convert.convert.ConvertOptionParser
print_version() (trans	<b>*</b> *
late.convert.po2wordfast.WfOptionParser	processfile() (trans-
method), 261	late.convert.po2moz.MozConvertOptionParser
<pre>print_version() (tran.</pre>	<i>''</i>
late.filters.pofilter.FilterOptionParser method	
350	late.convert.po2tmx.TmxOptionParser
<pre>print_version() (trans</pre>	**
late.misc.optrecurse.RecursiveOptionParser	processfile() (trans-
method), 392	late.convert.po2wordfast.WfOptionParser
<pre>print_version()</pre>	s- method), 261
late. tools. poconflicts. Conflict Option Parser	processfile() (trans-
method), 685	late.filters.pofilter.FilterOptionParser method),
print version() (tran.	s- 350

processfile() (trans-	method), 518
late.misc.optrecurse.RecursiveOptionParser method), 392	prune () (translate.storage.placeables.interfaces.InvisiblePlaceable method), 520
processfile() (trans-	prune() (translate.storage.placeables.interfaces.MaskingPlaceable
late.tools.poconflicts.ConflictOptionParser	method), 522
method), 685	<pre>prune() (translate.storage.placeables.interfaces.ReplacementPlaceable</pre>
processfile() (trans-	method), 523
late.tools.pogrep.GrepOptionParser method), 688	prune() (translate.storage.placeables.interfaces.SubflowPlaceable method), 525
processfile() (trans-	prune() (translate.storage.placeables.strelem.StringElem
late.tools.porestructure.SplitOptionParser	method), 528
method), 692	prune () (translate.storage.placeables.terminology.TerminologyPlaceable
processfile() (trans-	method), 530
late.tools.poterminology.TerminologyOptionPars method), 696	
394	prune() (translate.storage.placeables.xliff.Bx method), 536
Project (class in translate.storage.project), 567	prune() (translate.storage.placeables.xliff.Ept
ProjectStore (class in translate.storage.projstore),	method), 533
568	prune() (translate.storage.placeables.xliff.Ex method),
prop2inc() (in module trans-	538
late.convert.prop2mozfunny), 264	prune() (translate.storage.placeables.xliff.G method),
prop2it() (in module trans-	539
late.convert.prop2mozfunny), 264	prune() (translate.storage.placeables.xliff.It method),
prop2po (class in translate.convert.prop2po), 264	541
propertiesdecode() (in module trans-	prune() (translate.storage.placeables.xliff.Ph method),
late.misc.quote), 395	544
propfile (class in translate.storage.properties), 580	prune() (translate.storage.placeables.xliff.Sub
propunit (class in translate.storage.properties), 582	method), 543
prune () (translate.storage.placeables.base.Bpt	prune() (translate.storage.placeables.xliff.UnknownXML
method), 498	method), 546
	prune() (translate.storage.placeables.xliff.X method),
method), 506	535
	puncdict (translate.lang.common.Common attribute),
method), 500	359
	puncend () (in module translate.filters.decoration), 347
method), 508	puncspacing() (trans-
505	late.filters.checks.CCLicenseChecker method), 272
-	puncspacing() (trans-
503	late.filters.checks.DrupalChecker method),
prune() (translate.storage.placeables.base.Ph	278
method), 502	puncspacing() (trans-
prune() (translate.storage.placeables.base.Sub method), 511	late.filters.checks.GnomeChecker method), 284
prune () (translate.storage.placeables.base.X method), 510	puncspacing() (translate.filters.checks.IOSChecker method), 289
prune () (translate.storage.placeables.general.AltAttrPla	acpable spacing() (translate.filters.checks.KdeChecker method), 295
method), 513	
	ty Planeaty be cing () (translate.filters.checks.L20nChecker
method), 515	method), 301
prune () (translate.storage.placeables.general.XMLTagF	
method), 517	late.filters.checks.LibreOfficeChecker method),
prune () (translate.storage.placeables.interfaces.BasePla	aceable 307

puncspacing()		(trans-	method), 373		
late.filters.chec 312	cks.MinimalChecker	method),	<pre>punctranslate()     method), 374</pre>	(translate.lang.ko.ko	class
puncspacing()	ks.MozillaChecker	(trans- method),	punctranslate()  method), 374	(translate.lang.ml.ml	class
318	ns.moziiiaChecker	memou),	<pre>punctranslate()</pre>	(translate.lang.mr.mr	class
puncspacing()  late.filters.chec	cks.OpenOfficeChecker	(trans- method),	<pre>method), 375 punctranslate() method), 376</pre>	(translate.lang.ne.ne	class
puncspacing()	ks.ReducedChecker	(trans- method),	punctranslate()  method), 377	(translate.lang.pa.pa	class
330 puncspacing()		(trans-	punctranslate()(tr	anslate.lang.si.si class m	ethod),
	ks.StandardChecker	method),	punctranslate()(tr	anslate.lang.st.st class m	ethod),
puncspacing() (tran method), 342	ıslate.filters.checks.Terr	mChecker	punctranslate()  method), 380	(translate.lang.sv.sv	class
puncstart() (in mod 347	dule translate.filters.de	coration),	punctranslate()  method), 381	(translate.lang.ta.ta	class
<pre>punctranslate()     method), 353</pre>	(translate.lang.af.af	class	punctranslate()(tr	anslate.lang.te.te class m	ethod),
<pre>punctranslate()     method), 354</pre>	(translate.lang.am.an	ı class	<pre>punctranslate()     method), 382</pre>	(translate.lang.th.th	class
<pre>punctranslate()     method), 355</pre>	(translate.lang.bn.bn	class	<pre>punctranslate()     method), 383</pre>	(translate.lang.ug.ug	class
<pre>punctranslate() (translate.lang.code_or.code_or</pre>			punctranslate()  method), 384	(translate.lang.ur.ur	class
punctranslate() (  class method),	~	.Common	punctranslate()(tr	anslate.lang.vi.vi class m	ethod),
punctranslate()  method), 363	(translate.lang.de.de	class	<pre>punctranslate()     method), 385</pre>	(translate.lang.zh.zh	class
punctranslate()(t	ranslate.lang.el.el class	method),	punctuation (tatribute), 359	ranslate.lang.common.C	ommon
<pre>punctranslate()     method), 365</pre>	(translate.lang.es.es	class	purepunc () (translate, method), 272	filters.checks.CCLicense	Checker
<pre>punctranslate()     method), 366</pre>	(translate.lang.fa.fa	class	purepunc() (transle method), 278	ute.filters.checks.DrupalC	Checker
punctranslate()(t	ranslate.lang.fi.fi class	method),	purepunc() (transle method), 284	ute.filters.checks.GnomeC	Checker
punctranslate()(t	ranslate.lang.fr.fr class	method),	purepunc() (trainethod), 289	nslate.filters.checks.IOSC	Checker
<pre>punctranslate()     method), 368</pre>	(translate.lang.gu.gu	class	purepunc() (trainmethod), 295	nslate.filters.checks.KdeC	Checker
punctranslate()  method), 369	(translate.lang.he.he	class		slate.filters.checks.L20n <b>C</b>	Checker
<pre>punctranslate()     method), 370</pre>	(translate.lang.hi.hi	class	purepunc () (translate, method), 307	filters.checks.LibreOffice	Checker
punctranslate()  method), 370	(translate.lang.hy.hy	class	purepunc() (translat method), 312	e.filters.checks.MinimalC	Checker
punctranslate()  method), 371	(translate.lang.ja.ja	class		te.filters.checks.MozillaC	Checker
punctranslate()  method), 372	(translate.lang.km.km	ı class	purepunc() (translate. method), 324	filters.checks.OpenOffice	Checker
punctranslate()	(translate.lang.kn.kn	class	purepunc() (translat	e.filters.checks.Reduced0	Checker

method), 330	Q
purepunc() (translate.filters.checks.StandardChecker	qmfile (class in translate.storage.qm), 595
method), 335	qmunit (class in translate.storage.qm), 597
purepunc() (translate.filters.checks.TermChecker	qmunpack () (in module translate.storage.qm), 600
method), 343	QphFile (class in translate.storage.qph), 600
PyLuceneDatabase (class in trans-	QphUnit (class in translate.storage.qph), 602
late.search.indexing.PyLuceneIndexer), 400	QUERY_TYPE (translate.search.indexing.CommonIndexer.CommonDataba
PyLuceneHits (class in trans-	attribute), 397
late.search.indexing.PyLuceneIndexer), 402	quote_plus() (in module trans-
Python Enhancement Proposals	late.storage.pocommon), 557
PEP 257, 171	quoteforandroid() (in module trans-
PEP 8, 163, 165, 168	late.storage.dtd), 431
python_distance() (in module trans- late.search.lshtein), 405	quotefordtd() (in module translate.storage.dtd), 431
pythonbraceformat() (trans-	quoteforpo() (in module translate.storage.pypo),
late.filters.checks.CCLicenseChecker method),	594
272	quotes (translate.lang.common.Common attribute),
pythonbraceformat() (trans-	359
late.filters.checks.DrupalChecker method),	D
278	R
<pre>pythonbraceformat() (trans- late.filters.checks.GnomeChecker method),</pre>	rc2po (class in translate.convert.rc2po), 265
284	rcfile (class in translate.storage.rc), 606
pythonbraceformat() (trans-	RCS_METADIR (trans-
late.filters.checks.IOSChecker method), 290	late.storage.versioncontrol.GenericRevisionControlSystem
pythonbraceformat() (trans-	attribute), 662
late.filters.checks.KdeChecker method),	rcunit (class in translate.storage.rc), 607
295	read_obsolete_lines() (in module trans-
pythonbraceformat() (trans-	late.storage.poparser), 559
late.filters.checks.L20nChecker method),	read_prevmsgid_lines() (in module trans-
301	late.storage.poparser), 559
<pre>pythonbraceformat() (trans-</pre>	real_index() (in module translate.tools.pogrep), 689 reclassifyunit() (trans-
late.filters.checks.LibreOfficeChecker method),	late.storage.statistics.Statistics method),
307	611
pythonbraceformat() (trans-	recursearchivefiles() (trans-
late.filters.checks.MinimalChecker method),	late.convert.convert.ArchiveConvertOptionParser
313	method), 236
pythonbraceformat() (trans-	recursearchivefiles() (trans-
late.filters.checks.MozillaChecker method),	late.convert.po2tmx.TmxOptionParser
318	method), 256
pythonbraceformat() (trans-	recursearchivefiles() (trans-
late.filters.checks.OpenOfficeChecker method),	late.convert.po2wordfast.WfOptionParser
324	method), 261
pythonbraceformat() (trans-	recurseinputfilelist() (trans-
late.filters.checks.ReducedChecker method),	late.convert.convert.ArchiveConvertOptionParser
330	method), 236
pythonbraceformat() (trans- late.filters.checks.StandardChecker method),	recurseinputfilelist() (trans-
late.filters.checks.StandardChecker method), 336	late.convert.convert.ConvertOptionParser
pythonbraceformat() (trans-	method), 239
late.filters.checks.TermChecker method),	recurseinputfilelist() (trans-
343	late.convert.po2moz.MozConvertOptionParser
	method), 251
	recurseinputfilelist() (trans-
	late.convert.po2tmx.TmxOptionParser

method), 256		method), 696	
recurseinputfilelist()	(trans-	RecursiveOptionParser (class in	trans-
late.convert.po2wordfast.WfOptionPa	ırser	late.misc.optrecurse), 390	
method), 261		recursiveprocess()	(trans-
recurseinputfilelist()	(trans-	late.convert.convert.ArchiveConvertOp	tionParser
late. filters. pofilter. Filter Option Parser	· method),	method), 236	
350		recursiveprocess()	(trans-
recurseinputfilelist()	(trans-	late.convert.convert.ConvertOptionPars	ser
late.misc.optrecurse.RecursiveOption	Parser	method), 239	
method), 392		recursiveprocess()	(trans-
recurseinputfilelist()	(trans-	late.convert.po2moz.MozConvertOption	ıParser
late.tools.poconflicts.ConflictOptionF	Parser	method), 251	
method), 685		recursiveprocess()	(trans-
recurseinputfilelist()	(trans-	late.convert.po2tmx.TmxOptionParser	
late.tools.pogrep.GrepOptionParser	method),	method), 256	
689		recursiveprocess()	(trans-
recurseinputfilelist()	(trans-	late.convert.po2wordfast.WfOptionPars	ser
late.tools.porestructure.SplitOptionPo	arser	method), 261	,
method), 692	,	recursiveprocess()	(trans-
recurseinputfilelist()	(trans-	late.filters.pofilter.FilterOptionParser 1	nethod),
late.tools.poterminology.Terminology	OptionPars		
method), 696		recursiveprocess()	(trans-
recurseinputfiles()	(trans-	late.misc.optrecurse.RecursiveOptionPo	arser
late.convert.convert.ArchiveConvertC	)ptionParse		(4
method), 236	(4	recursiveprocess()	(trans-
recurseinputfiles()	(trans-	late.tools.poconflicts.ConflictOptionPar	rser
late.convert.convert.ConvertOptionPo	irser	method), 685	(tuana
method), 239	(tuans	recursiveprocess()	(trans-
recurseinputfiles()	(trans-	late.tools.pogrep.GrepOptionParser 1 689	neinoa),
late.convert.po2moz.MozConvertOpti method), 251	ionrarser		(trans-
recurseinputfiles()	(trans-	recursiveprocess()	
late.convert.po2tmx.TmxOptionParse		late.tools.porestructure.SplitOptionPar. method), 692	ser
method), 256	,	recursiveprocess()	(trans-
recurseinputfiles()	(trans-	late.tools.poterminology.TerminologyO	`
late.convert.po2wordfast.WfOptionPa	•	method), 696	puom arse.
method), 261	ii sei	redtd (class in translate.convert.po2dtd), 246	
recurseinputfiles()	(trans-	reduce_tree() (in module	trans-
late.filters.pofilter.FilterOptionParser		late.storage.xml_extract.misc), 680	
350	,	ReducedChecker (class in translate.filters.	checks).
recurseinputfiles()	(trans-	327	,,
late.misc.optrecurse.RecursiveOption	,	register_dialect() (in module	trans-
method), 392		late.storage.ini), 451	
recurseinputfiles()	(trans-	register_dialect() (in module	trans-
late.tools.poconflicts.ConflictOptionF	Parser	late.storage.properties), 584	
method), 685		reindent() (in module translate.misc.xml_l	helpers),
recurseinputfiles()	(trans-	396	1 //
late.tools.pogrep.GrepOptionParser	method),	remove_file()	(trans-
689	**	late.storage.bundleprojstore.BundlePro	•
recurseinputfiles()	(trans-	method), 414	
late.tools.porestructure.SplitOptionPo		remove_file() (translate.storage.project	t.Project
method), 692		method), 567	-
recurseinputfiles()	(trans-	remove_file()	(trans-
late.tools.poterminology.Terminology	OptionPars	er late.storage.projstore.ProjectStore i	nethod),

	568			method), 530	
remove_	_spreadsheet_escapes()	(trans-	remove	_type()	(trans-
	late.storage.csvl10n.csvunit method),	424		late.storage.placeables.xliff.Bpt	method),
remove_	_type()	(trans-		531	
	late.storage.placeables.base.Bpt 498	method),	remove	_type() (translate.storage.placeab method), 536	les.xliff.Bx
remove_	_type()(translate.storage.placeable	es.base.Bx	remove	_type()	(trans-
	method), 507			late.storage.placeables.xliff.Ept	method),
remove_	_type()	(trans-		533	
	late.storage.placeables.base.Ept 500	method),	remove_	_type() (translate.storage.placeab method), 538	les.xliff.Ex
remove_	_type() (translate.storage.placeable method), 508	es.base.Ex	remove_	_type() (translate.storage.placeae method), 540	bles.xliff.G
remove_	_type() (translate.storage.placeable method), 505	les.base.G	remove	_type() (translate.storage.placea method), 541	bles.xliff.It
remove_	_type() (translate.storage.placeab method), 503	les.base.It	remove_	_type() (translate.storage.placeab method), 544	les.xliff.Ph
remove	_type()	(trans-	remove	_type()	(trans-
	late.storage.placeables.base.Ph 502	method),		late.storage.placeables.xliff.Sub 543	method).
remove_	_type()	(trans-	remove	_type()	(trans-
	late.storage.placeables.base.Sub 511	method),		late.storage.placeables.xliff.Unknown method), 546	nXML
remove_	_type() (translate.storage.placeab method), 510	les.base.X	remove_	_type() (translate.storage.placea method), 535	bles.xliff.X
remove_	_type()	(trans-	remove	_unit_from_index()	(trans-
	late.storage.placeables.general.AltAtt method), 513	trPlaceable		late.storage.base.TranslationStore 409	method),
remove_	_type()	(trans-	remove	_unit_from_index()	(trans-
	late.storage.placeables.general.XML1 method), 515	EntityPlacea	ıble	late.storage.catkeys.CatkeysFile 416	method),
remove_	_type()	(trans-	remove	_unit_from_index()	(trans-
	late.storage.placeables.general.XML1 method), 517	TagPlaceabl		<pre>late.storage.csvl10n.csvfile method), _unit_from_index()</pre>	421 (trans-
remove_	_type()	(trans-		late.storage.dtd.dtdfile method), 427	
	late. storage. placeables. interfaces. Baseline and the storage of the storage	sePlaceable	remove		(trans-
	method), 518			late.storage.html.htmlfile method), 43	37
remove_	_type()			_unit_from_index()	(trans-
	late.storage.placeables.interfaces.Inv method), 520			late.storage.html.POHTMLParser 434	method),
remove_	_type()			_unit_from_index()	(trans-
	late.storage.placeables.interfaces.Ma method), 522	, and the second		<pre>late.storage.ical.icalfile method), 442 _unit_from_index()</pre>	(trans-
remove_	_type()	(trans-		late.storage.ini.inifile method), 447	4.
	late.storage.placeables.interfaces.Rep method), 523		a <b>ceabl</b> ⊌e.	late. storage. js on 110 n. I18 Next File	(trans- method),
remove_	_type()	(trans-		453	
	late.storage.placeables.interfaces.Sub	oftowPlaceal	<i>bte</i> emove_		(trans-
	method), 525	,		late.storage.jsonl10n.JsonFile metho	
remove_	_type()		remove_	_unit_from_index()	(trans-
	late.storage.placeables.strelem.String method), 528			late.storage.jsonl10n.JsonNestedFile 459	
remove_	_type() late.storage.placeables.terminology.T			_unit_from_index() late.storage.jsonl10n.WebExtensionJ	(trans- sonFile

method), 467	617
<pre>remove_unit_from_index() (trans-</pre>	remove_unit_from_index() (trans-
late.storage.lisa.LISAfile method), 472	late.storage.subtitles.SubStationAlphaFile
remove_unit_from_index() (trans-	method), 618
late.storage.mo.mofile method), 478	remove_unit_from_index() (trans-
remove_unit_from_index() (trans-	late.storage.subtitles.SubtitleFile method),
late.storage.mozilla_lang.LangStore method),	620
483	remove_unit_from_index() (trans-
remove_unit_from_index() (trans-	late.storage.tbx.tbxfile method), 625
late.storage.omegat.OmegaTFile method),	remove_unit_from_index() (trans-
489	late.storage.tiki.TikiStore method), 631
remove_unit_from_index() (trans-	remove_unit_from_index() (trans-
late.storage.omegat.OmegaTFileTab method), 490	late.storage.tmx.tmxfile method), 636
	remove_unit_from_index() (trans-
remove_unit_from_index() (trans-	late.storage.trados.TradosTxtTmFile method),
late.storage.php.phpfile method), 548	644
remove_unit_from_index() (trans-	remove_unit_from_index() (trans-
late.storage.pocommon.pofile method), 554	late.storage.ts2.tsfile method), 647
remove_unit_from_index() (trans-	remove_unit_from_index() (trans-
late.storage.poxliff.PoXliffFile method), 562	late.storage.txt.TxtFile method), 652
remove_unit_from_index() (trans-	remove_unit_from_index() (trans-
late.storage.properties.javafile method),	late.storage.utx.UtxFile method), 657
576	remove_unit_from_index() (trans-
<pre>remove_unit_from_index() (trans-</pre>	$late.storage.word fast.Word fast TMFile\ method),$
late.storage.properties.javautf8file method),	668
578	remove_unit_from_index() (trans-
<pre>remove_unit_from_index() (trans-</pre>	late.storage.xliff.xlifffile method), 674
late.storage.properties.joomlafile method),	removedefaultfile() (trans-
579	late.storage.poxliff.PoXliffFile method), 562
remove_unit_from_index() (trans-	removedefaultfile() (trans-
late.storage.properties.propfile method),	late.storage.xliff.xlifffile method), 674
581	removeduplicates() (translate.storage.pypo.pofile
remove_unit_from_index() (trans-	method), 590
late.storage.properties.stringsfile method),	
586 tate.storage.properties.stringsjae methoa),	
	late.storage.dtd), 431
remove_unit_from_index() (trans-	removekdecomments() (in module trans-
late.storage.properties.stringsutf8file method),	late.filters.prefilters), 352
587	removenotes() (trans-
remove_unit_from_index() (trans-	late.storage.base.TranslationUnit method),
late.storage.pypo.pofile method), 590	412
remove_unit_from_index() (trans-	removenotes() (trans-
late.storage.qm.qmfile method), 596	late.storage.catkeys.CatkeysUnit method),
remove_unit_from_index() (trans-	419
late.storage.qph.QphFile method), 602	removenotes() (translate.storage.csvl10n.csvunit
<pre>remove_unit_from_index() (trans-</pre>	method), 424
late.storage.rc.rcfile method), 607	removenotes() (translate.storage.dtd.dtdunit
remove_unit_from_index() (trans-	method), 430
late.storage.subtitles.AdvSubStationAlphaFile	removenotes() (translate.storage.html.htmlunit
method), 613	method), 440
remove_unit_from_index() (trans-	removenotes() (translate.storage.ical.icalunit
late.storage.subtitles.MicroDVDFile method),	method), 445
615	removenotes () (translate.storage.ini.iniunit method),
remove_unit_from_index() (trans-	450
late.storage.subtitles.SubRipFile method),	removenotes() (trans-
include),	truis

late.storage.jsonl10n.I18NextUnit method), 456	removenotes() (trans- late.storage.wordfast.WordfastUnit method),
removenotes() (trans- late.storage.jsonl10n.JsonNestedUnit method),	671 removenotes() (translate.storage.xliff.xliffunit
462	method), 678
removenotes () (translate.storage.jsonl10n.JsonUnit method), 465	renderer (translate.storage.placeables.strelem.StringElem attribute), 528
removenotes() (trans-	replace() (translate.misc.multistring.multistring
late.storage.jsonl10n.WebExtensionJsonUnit	method), 389
method), 470	replace_dom_text() (in module trans-
removenotes () (translate.storage.lisa.LISAunit	late.storage.xml_extract.generate), 679
method), 475	ReplacementPlaceable (class in trans- late.storage.placeables.interfaces), 522
removenotes () (translate.storage.mo.mounit method), 481	Replacer (class in translate.convert.convert), 240
removenotes() (trans-	replacestrings() (in module trans-
late.storage.mozilla_lang.LangUnit method),	late.convert.csv2po), 241
486	require_index() (trans-
removenotes() (trans-	late.storage.base.TranslationStore method),
late.storage.omegat.OmegaTUnit method),	409
493	require_index() (trans-
removenotes() (translate.storage.php.phpunit method), 551	late.storage.catkeys.CatkeysFile method), 416
removenotes() (translate.storage.pocommon.pounit method), 557	require_index() (translate.storage.csvl10n.csvfile method), 421
removenotes () (translate.storage.poxliff.PoXliffUnit method), 566	require_index() (translate.storage.dtd.dtdfile method), 427
removenotes() (trans-	require_index() (translate.storage.html.htmlfile
late.storage.properties.propunit method),	method), 437
584	require_index() (trans-
removenotes() (translate.storage.pypo.pounit method), 593	late.storage.html.POHTMLParser method), 434
removenotes () (translate.storage.qm.qmunit method), 599	require_index() (translate.storage.ical.icalfile method), 442
removenotes () (translate.storage.qph.QphUnit method), 605	require_index() (translate.storage.ini.inifile method), 447
removenotes() (translate.storage.rc.rcunit method),	require_index() (trans-
610	late.storage.jsonl10n.I18NextFile method),
removenotes() (trans-	453
late.storage.subtitles.SubtitleUnit method),	require_index() (trans-
623	late.storage.jsonl10n.JsonFile method), 457
removenotes() (translate.storage.tbx.tbxunit	require_index() (trans-
method), 628	late. storage. js on 110 n. Js on Nested File  method),
removenotes() (translate.storage.tiki.TikiUnit	459
method), 633	require_index() (trans-
removenotes() (translate.storage.tmx.tmxunit	late. storage. js on 110 n. Web Extension Js on File
method), 639	method), 467
removenotes () (translate.storage.trados.TradosUnit method), 643	require_index() (translate.storage.lisa.LISAfile method), 472
${\tt removenotes()} \ \ \textit{(translate.storage.ts2.tsunit method)}, \\ 650$	require_index() (translate.storage.mo.mofile method), 478
removenotes() (translate.storage.txt.TxtUnit	require_index() (trans-
method), 655	$late.storage.mozilla\_lang.LangStore  method),$
removenotes() (translate.storage.utx.UtxUnit	483
method), 661	require_index() (trans-

late.storage.omegat.OmegaTFile m 489	ethod),	require_index(  method), 6	=
require_index()	(trans-	require_index(	) (translate.storage.tmx.tmxfile
late.storage.omegat.OmegaTFileTab m 490	ethod),	<pre>method), 6 require_index(</pre>	
	nhnfila	_	
require_index() (translate.storage.php. method), 548	.рпрјие	644	e.trados.TradosTxtTmFile method),
<pre>require_index()     late.storage.pocommon.pofile method),5</pre>	(trans- 554	require_index(  method), 6	
require_index()  late.storage.poxliff.PoXliffFile method),	(trans-	require_index(  method), 6	) (translate.storage.txt.TxtFile
	(trans-	require_index(	
late.storage.properties.javafile m	ethod),	method), 6	57
576		require_index(	
require_index()	(trans-	_	$e.word fast. Word fast TMF ile\ method),$
~	ethod),	668	
578	,	require_index(	
require_index()	(trans-	method), 6	
late.storage.properties.joomlafile m 579	ethod),	reset() (tra method), 3	nslate.misc.ourdom.ExpatBuilderNS 93
require_index()	(trans-	reset()(translate	.storage.html.htmlfile method), 437
late.storage.properties.propfile m 581	ethod),	reset() (tra method), 4	inslate.storage.html.POHTMLParser 34
require_index()	(trans-	· · · · · · · · · · · · · · · · · · ·	unslate.storage.pypo.pounit method),
	ethod),	594	
586	,,	resx2po (class in t	ranslate.convert.resx2po), 265
require_index()	(trans-		ranslate.misc.multistring.multistring
late.storage.properties.stringsutf8file m	ethod),	method), 3	
587		rich_parsers	(trans-
require_index() (translate.storage.pype method), 590	o.pofile	late.storage 412	e.base.TranslationUnit attribute),
require_index() (translate.storage.qm method), 596	ı.qmfile	rich_source (transcribute),	anslate.storage.base.TranslationUnit 412
require_index() (translate.storage.qph.Q method), 602	QphFile		ranslate.storage.catkeys.CatkeysUnit
require_index() (translate.storage.nethod), 607	rc.rcfile	rich_source  attribute),	(translate. storage. csvl 10 n. csvunit
require_index()  late.storage.subtitles.AdvSubStationAlph	(trans-	* * * * * * * * * * * * * * * * * * * *	nslate.storage.dtd.dtdunit attribute),
method), 613	icii iic		translate.storage.html.htmlunit at-
require_index()	(trans-	<i>tribute</i> ), 44	_
late.storage.subtitles.MicroDVDFile m	ethod),	rich_source (	translate.storage.ical.icalunit at-
615	(tuans	tribute), 44	
require_index()  late.storage.subtitles.SubRipFile m	(trans- ethod),	450	anslate.storage.ini.iniunit attribute),
617	einou),		nslate.storage.jsonl10n.I18NextUnit
require_index()	(trans-	attribute),	
late.storage.subtitles.SubStationAlphaFi	le	rich_source	(trans-
method), 618	,	_	e.jsonl10n.JsonNestedUnit at-
require_index()	(trans-	tribute), 46	
late.storage.subtitles.SubtitleFile m 620	ethod),	rich_source (tra tribute), 46	nslate.storage.jsonl10n.JsonUnit at- 5
require_index() (translate.storage.tbx method), 625	x.tbxfile	rich_source late.storage	(trans- e.jsonl10n.WebExtensionJsonUnit

- attribute), 470 (translate.storage.lisa.LISAunit rich source tribute), 475 rich\_source (translate.storage.mo.mounit attribute), 481 rich source (translate.storage.mozilla lang.LangUnit attribute), rich\_source (translate.storage.omegat.OmegaTUnit attribute), 493 rich\_source (translate.storage.php.phpunit attribute), 551 rich\_source (translate.storage.pocommon.pounit attribute), 557 rich\_source (translate.storage.poxliff.PoXliffUnit attribute), 566 (translate.storage.properties.propunit rich\_source attribute), 584 rich\_source (translate.storage.pypo.pounit attribute), 594 rich\_source (translate.storage.qm.qmunit attribute), 599 rich\_source (translate.storage.qph.QphUnit tribute), 605 rich\_source (translate.storage.rc.rcunit attribute), 610 rich\_source (translate.storage.subtitles.SubtitleUnit attribute), 623 rich\_source (translate.storage.tbx.tbxunit attribute), 628 rich\_source (translate.storage.tiki.TikiUnit tribute), 634 rich\_source (translate.storage.tmx.tmxunit tribute), 639 rich\_source (translate.storage.trados.TradosUnit at*tribute*), 643 rich source (translate.storage.ts2.tsunit attribute), rich\_source (translate.storage.txt.TxtUnit attribute), 655 (translate.storage.utx.UtxUnit rich source attribute), 661 rich source (translate.storage.wordfast.WordfastUnit attribute), rich\_target (translate.storage.base.TranslationUnit attribute), 412 rich\_target (translate.storage.catkeys.CatkeysUnit attribute), 419 (translate.storage.csvl10n.csvunit rich\_target attribute), 424 rich\_target (translate.storage.dtd.dtdunit attribute), rich\_target (translate.storage.txt.TxtUnit attribute), 430 rich target (translate.storage.html.htmlunit 655
- tribute), 440 (translate.storage.ical.icalunit rich\_target attribute), 445 rich\_target (translate.storage.ini.iniunit attribute), rich target (translate.storage.jsonl10n.I18NextUnit attribute), 456 rich target (translate.storage.jsonl10n.JsonNestedUnit attribute), 462 rich\_target (translate.storage.jsonl10n.JsonUnit at*tribute*), 465 (transrich\_target late.storage.jsonl10n.WebExtensionJsonUnit attribute), 470 rich\_target (translate.storage.lisa.LISAunit *tribute*), 476 rich\_target (translate.storage.mo.mounit attribute), 482 rich\_target (translate.storage.mozilla\_lang.LangUnit attribute), rich\_target (translate.storage.omegat.OmegaTUnit attribute), 494 rich\_target (translate.storage.php.phpunit attribute), 551 rich\_target (translate.storage.pocommon.pounit attribute), 557 rich\_target (translate.storage.poxliff.PoXliffUnit at*tribute*), 566 rich\_target (translate.storage.properties.propunit attribute), 584 rich\_target (translate.storage.pypo.pounit tribute), 594 rich\_target (translate.storage.qm.qmunit attribute), rich target (translate.storage.qph.QphUnit *tribute*), 605 rich\_target (translate.storage.rc.rcunit attribute), 610 rich target (translate.storage.subtitles.SubtitleUnit attribute), 623 rich\_target (translate.storage.tbx.tbxunit attribute), 629 rich\_target (translate.storage.tiki.TikiUnit attribute), 634 rich\_target (translate.storage.tmx.tmxunit at*tribute*), 639 rich\_target (translate.storage.trados.TradosUnit at*tribute*), 643 rich\_target (translate.storage.ts2.tsunit attribute),

<pre>rich_target (translate.storage.utx.UtxUnit at-</pre>	late.storage.poxliff.PoXliffUnit class method), 566
rich_target (trans- late.storage.wordfast.WordfastUnit attribute), 671	rich_to_multistring() (trans- late.storage.properties.propunit class method), 584
rich_to_multistring() (trans-	rich_to_multistring() (trans-
late.storage.base.TranslationUnit class method), 413	late.storage.pypo.pounit class method), 594
rich_to_multistring() (trans-	rich_to_multistring() (trans-
late.storage.catkeys.CatkeysUnit class method),	late.storage.qm.qmunit class method), 599
419	rich_to_multistring() (trans-
	late.storage.qph.QphUnit class method), 605
424	rich_to_multistring() (trans-
rich_to_multistring() (trans-	late.storage.rc.rcunit class method), 610
late.storage.dtd.dtdunit class method), 430	rich_to_multistring() (trans-
rich_to_multistring() (trans-	late.storage.subtitles.SubtitleUnit class
late.storage.html.htmlunit class method),	method), 623
440	rich_to_multistring() (trans-
rich_to_multistring() (trans-	late.storage.tbx.tbxunit class method), 629
late.storage.ical.icalunit class method), 445	rich_to_multistring() (trans- late.storage.tiki.TikiUnit class method),
rich_to_multistring() (trans-	634
late.storage.ini.iniunit class method), 450	rich_to_multistring() (trans-
rich_to_multistring() (trans-	late.storage.tmx.tmxunit class method), 639
late.storage.jsonl10n.I18NextUnit class method), 456	rich_to_multistring() (trans- late.storage.trados.TradosUnit class method),
rich_to_multistring() (trans-	643
late.storage.jsonl10n.JsonNestedUnit class method), 462	rich_to_multistring() (trans- late.storage.ts2.tsunit class method), 650
rich_to_multistring() (trans-	rich_to_multistring() (trans-
late.storage.jsonl10n.JsonUnit class method), 465	late.storage.txt.TxtUnit class method), 655 rich_to_multistring() (trans-
rich_to_multistring() (trans-	late.storage.utx.UtxUnit class method), 661
late.storage.jsonl10n.WebExtensionJsonUnit class method), 470	rich_to_multistring() (trans- late.storage.wordfast.WordfastUnit class
rich_to_multistring() (trans-	method), 671
late.storage.lisa.LISAunit class method),	rich_to_multistring() (trans-
476	late.storage.xliff.xliffunit class method),
rich_to_multistring() (trans-	678
late.storage.mo.mounit class method), 482	rindex() (translate.misc.multistring.multistring
rich_to_multistring() (trans- late.storage.mozilla_lang.LangUnit class	method), 389 rjust() (translate.misc.multistring.multistring
method), 487	method), 389
rich_to_multistring() (trans-	<pre>rpartition() (translate.misc.multistring.multistring</pre>
late.storage.omegat.OmegaTUnit class	method), 389
method), 494	rsplit() (translate.misc.multistring.multistring
rich_to_multistring() (trans-	method), 389
late.storage.php.phpunit class method), 551	rstrip() (translate.misc.multistring.multistring method), 389
rich_to_multistring() (trans-	RTF_ESCAPES (in module translate.storage.trados),
late.storage.pocommon.pounit class method),	640
557	rtlpunc (translate.lang.common.Common attribute),
rich_to_multistring() (trans-	359

run()	(translate.convert	.convert.Ar	chiveConv	ertOptionPa	ırser	late.conver	t.po2mozl	ang), 248	
	method), 236				run_co	nverter(	) (in	ı modu	le trans-
run()	(translate.conve	rt.convert.	ConvertOp	tionParser		late.conver	t.po2tiki),	254	
	method), 239				run_co	nverter(	) (in	ı modu	ele trans-
	(translate.convert					late.conver	t.po2txt),	258	
	(translate.convert	-	-		run_co	nverter(	) (ir	ı modu	ele trans-
run()	(translate.convert	t.mozlang2 <sub>l</sub>	po.lang2po	o method),		late.conver	t.po2yaml	), 263	
	245				run_co	nverter(	) (ir	ı modu	ele trans-
	(translate.convert					late.conver		266	
	(translate.convert				run_co	nverter(			ele trans-
	(translate.convert					late.conver			
run()	(translate.convert	.po2moz.M	ozConvert	<i>OptionParse</i>	errun_co				ele trans-
	method), 251					late.conver	t.yaml2po	), 268	
run()	(translate.convert	t.po2mozlai	ng.po2lang	g method),	run_fi	lters()			(trans-
	248						checks.CC	LicenseChe	cker method),
	(translate.convert	_				272			
run()	(translate.co	nvert.po2tr	пх.ТтхОр	tionParser	run_fi	lters()			(trans-
	method), 256						checks.Dri	upalChecker	method),
	(translate.convert					278			
run()	(translate.conve	rt.po2word	lfast.WfOp	tionParser	run_fi	lters()			(trans-
	method), 261						checks.Gn	omeChecker	method),
run()	(translate.conve	ert.po2yaml	l.po2yaml	method),		284			
	263				run_fi			filters.check	ks.IOSChecker
	(translate.convert	_	_			method), 29			
	(translate.convert	_	_		run_fi			.filters.check	xs.KdeChecker
run()	(translate.conve	ert.yaml2po	yaml2po	method),		method), 29		C1. 1 1	100 01 1
	268		T-11 0		run_fi			hlters.checks	.L20nChecker
run()	(translate.fi	lters.pofilte	er.FilterOp	tionParser		method), 30	)1		
	method), 350			· D	run_fi	lters()	1 1 7 . 7	0.66 (7)	(trans-
run()	(translate.misc.op method), 392	trecurse.Re	ecursive <b>O</b> p	otionParser		late.filters.c	checks.Lib	reOfficeChe	cker method),
run()	(translate.tools.p	oconflicts.	ConflictOp	tionParser	run fi	lters()			(trans-
- ( /	method), 685	<i>J</i>	$J \cdots I$		-		checks.Mi	nimalChecke	,
run()		tools.pogre	p.GrepOp	tionParser		313			,,,
- ( /	method), 689	7 - 6	r · · · r		run fi	lters()			(trans-
run()	(translate.tools.)	porestructu	re.SplitOp	tionParser	-		checks.Mo	zillaChecke	*
.,	method), 692		1 1			318			,,
run()	(translate.tools.pd	oterminolog	y.Termino	logyOption <b>l</b>	Panser fi	lters()			(trans-
.,	method), 696		,,,	0.7	_		checks.Op	enOfficeChe	cker method),
run c	ommand()	(in	module	trans-		324	1	33	,,
_	late.storage.ver	*			run fi	lters()			(trans-
run c	onverter()	(in	module	trans-	_		checks.Red	ducedChecke	,
_	late.convert.ica	`				330			,,
run c	onverter()	(in	module	trans-	run fi	lters()			(trans-
_	late.convert.ini2	2po), 243			_		checks.Sta	ndardCheck	er method),
run c	onverter()	(in	module	trans-		336			,,
_	late.convert.mo	zlang2po),	245		run fi	lters()			(trans-
run c	onverter()	(in	module	trans-	_		checks.Sta	ndardUnitC	hecker
	late.convert.php	`				method), 33			
run c	onverter()	(in	module	trans-	run fi			e.filters.chec	ks.TeeChecker
	late.convert.po2	`				method), 33			
run_c	onverter()	(in	module	trans-	run_fi			filters.checks	.TermChecker
_	late.convert.po2	`			_	method), 34			
run_c	onverter()	(in	module	trans-	run_fi	lters()			(trans-

late.filters.checks.TranslationChecker method), 345	save()	(translate.storage.html.POHTMLParser method), 434
<pre>run_filters() (translate.filters.checks.UnitChecker</pre>	save()	(translate.storage.ical.icalfile method), 442
method), 346		(translate.storage.ini.inifile method), 447
run_test() (translate.filters.checks.CCLicenseChecker		(translate.storage.jsonl10n.I18NextFile
method), 272	54.5()	method), 453
run_test() (translate.filters.checks.DrupalChecker	save ()	(translate.storage.jsonl10n.JsonFile method),
method), 278	save ()	458
	00TTO ()	
	save()	(translate.storage.jsonl10n.JsonNestedFile method), 459
method), 284	()	
run_test() (translate.filters.checks.IOSChecker	save()	(translate.storage.jsonl10n.WebExtensionJsonFile
method), 290		method), 467
run_test() (translate.filters.checks.KdeChecker		(translate.storage.lisa.LISAfile method), 472
method), 295		(translate.storage.mo.mofile method), 478
run_test() (translate.filters.checks.L20nChecker	save()	$(translate.storage.mozilla\_lang.LangStore$
method), 301		method), 483
$\verb"run_test"()" (\textit{translate.filters.checks.LibreOfficeChecker"})$	save()	(translate.storage.omegat.OmegaTFile method),
method), 307		489
<pre>run_test() (translate.filters.checks.MinimalChecker</pre>	save()	(translate. storage. omegat. OmegaTF ile Tab
method), 313		method), 490
<pre>run_test() (translate.filters.checks.MozillaChecker</pre>	save()	(translate.storage.php.phpfile method), 548
method), 318	save()	(translate.storage.pocommon.pofile method),
<pre>run_test() (translate.filters.checks.OpenOfficeChecker</pre>		554
method), 324	save()	(translate.storage.poxliff.PoXliffFile method),
<pre>run_test() (translate.filters.checks.ReducedChecker</pre>		562
method), 330	save()	(translate.storage.project.Project method), 568
run_test() (translate.filters.checks.StandardChecker	save()	(translate.storage.projstore.ProjectStore
method), 336	54.5()	method), 568
run_test() (translate.filters.checks.StandardUnitChecke	erave ()	
method), 339	bave ()	576
run_test() (translate.filters.checks.TermChecker	save()	(translate.storage.properties.javautf8file
method), 343	save ()	method), 578
	0.0110 ()	
run_test() (translate.filters.checks.TranslationChecker	save()	(translate.storage.properties.joomlafile
method), 345	()	method), 579
	save()	(translate.storage.properties.propfile method),
method), 346	()	581
runclean() (in module translate.tools.poclean), 682	save()	(translate.storage.properties.stringsfile
runfilter() (in module translate.filters.pofilter), 351		method), 586
rungrep () (in module translate.tools.pogrep), 690	save()	(translate.storage.properties.stringsutf8file
runtests() (in module translate.filters.checks), 346		method), 587
0		(translate.storage.pypo.pofile method), 590
S		(translate.storage.qm.qmfile method), 596
<pre>safe_escape() (in module translate.storage.html),</pre>	save()	(translate.storage.qph.QphFile method), 602
440	save()	(translate.storage.rc.rcfile method), 607
save() (translate.storage.base.TranslationStore	save()	(translate. storage. subtitles. Adv SubStation Alpha File
method), 409		method), 613
save() (translate.storage.bundleprojstore.BundleProjectS	save ()	(translate.storage.subtitles.MicroDVDFile
method), 414		method), 615
save() (translate.storage.catkeys.CatkeysFile method),	save()	(translate.storage.subtitles.SubRipFile method),
416	.,	617
save() (translate.storage.csvl10n.csvfile method), 421	save()	(translate.storage.subtitles.SubStationAlphaFile
save () (translate.storage.dtd.dtdfile method), 427	()	method), 619
save () (translate.storage.ata.atajae method), 427 save () (translate.storage.html.htmlfile method), 437	save()	(translate.storage.subtitles.SubtitleFile
save () (transtate.storage.nimi.nimijue memoa), 43/	()	method) 620

```
save () (translate.storage.tbx.tbxfile method), 625
                                                        savefile()
                                                                           (translate.storage.properties.propfile
save () (translate.storage.tiki.TikiStore method), 631
                                                                 method), 581
save () (translate.storage.tmx.tmxfile method), 636
                                                        savefile()
                                                                         (translate.storage.properties.stringsfile
              (translate.storage.trados.TradosTxtTmFile
                                                                 method), 586
save()
         method), 645
                                                        savefile() (translate.storage.properties.stringsutf8file
save () (translate.storage.ts2.tsfile method), 647
                                                                 method), 588
save () (translate.storage.txt.TxtFile method), 652
                                                        savefile()
                                                                       (translate.storage.pypo.pofile method),
save () (translate.storage.utx.UtxFile method), 657
                                                                 590
save()
            (translate.storage.wordfast.WordfastTMFile
                                                        savefile() (translate.storage.qm.qmfile method), 596
         method), 668
                                                        savefile() (translate.storage.qph.QphFile method),
save() (translate.storage.xliff.xlifffile method), 674
                                                                 602
savefile() (translate.storage.base.TranslationStore
                                                        savefile() (translate.storage.rc.rcfile method), 607
         method), 409
                                                        savefile() (translate.storage.subtitles.AdvSubStationAlphaFile
savefile()
                  (translate.storage.catkeys.CatkeysFile
                                                                 method), 614
                                                        savefile() (translate.storage.subtitles.MicroDVDFile
         method), 416
savefile()
                       (translate.storage.csvl10n.csvfile
                                                                 method), 615
         method), 421
                                                        savefile()
                                                                         (translate.storage.subtitles.SubRipFile
savefile() (translate.storage.dtd.dtdfile method), 427
                                                                 method), 617
savefile() (translate.storage.html.htmlfile method),
                                                        savefile() (translate.storage.subtitles.SubStationAlphaFile
         437
                                                                 method), 619
savefile()
                (translate.storage.html.POHTMLParser
                                                        savefile()
                                                                         (translate.storage.subtitles.SubtitleFile
         method), 434
                                                                 method), 620
                                                        savefile() (translate.storage.tbx.tbxfile method), 625
savefile()
               (translate.storage.ical.icalfile method),
                                                        savefile() (translate.storage.tiki.TikiStore method),
savefile() (translate.storage.ini.inifile method), 447
                                                                 631
savefile()
                (translate.storage.jsonl10n.I18NextFile
                                                       savefile()
                                                                       (translate.storage.tmx.tmxfile method),
                                                                 636
         method), 453
                                                        savefile() (translate.storage.trados.TradosTxtTmFile
savefile()
                    (translate.storage.jsonl10n.JsonFile
         method), 458
                                                                 method), 645
savefile()(translate.storage.jsonl10n.JsonNestedFile savefile()(translate.storage.ts2.tsfile method), 647
         method), 459
                                                        savefile()
                                                                       (translate.storage.txt.TxtFile method),
savefile() (translate.storage.jsonl10n.WebExtensionJsonFile
                                                                 652
         method), 467
                                                        savefile()
                                                                       (translate.storage.utx.UtxFile method),
                                                                 657
savefile() (translate.storage.lisa.LISAfile method),
         472
                                                        savefile() (translate.storage.wordfast.WordfastTMFile
savefile() (translate.storage.mo.mofile method), 478
                                                                 method), 668
savefile()(translate.storage.mozilla lang.LangStore
                                                        savefile() (translate.storage.xliff.xlifffile method),
         method), 483
                                                                 674
savefile()
                 (translate.storage.omegat.OmegaTFile
                                                        SCAN PARENTS
                                                                                                       (trans-
         method), 489
                                                                 late. storage. version control. Generic Revision Control System\\
savefile() (translate.storage.omegat.OmegaTFileTab
                                                                 attribute), 662
         method), 490
                                                                          (translate.storage.directory.Directory
                                                        scanfiles()
savefile() (translate.storage.php.phpfile method),
                                                                 method), 425
         548
                                                        scanfiles() (translate.storage.zip.ZIPFile method),
savefile()
                    (translate.storage.pocommon.pofile
         method), 554
                                                        scripts (in module translate.lang.data), 361
savefile()
                    (translate.storage.poxliff.PoXliffFile
                                                        search() (translate.search.indexing.CommonIndexer.CommonDatabase
         method), 562
                                                                 method), 399
savefile()
                   (translate.storage.properties.javafile
                                                        search() (translate.search.indexing.PyLuceneIndexer.PyLuceneDatabase
         method), 576
                                                                 method), 401
                                                        search() (translate.search.indexing.XapianIndexer.XapianDatabase
savefile() (translate.storage.properties.javautf8file
                                                                 method), 404
savefile()
                 (translate.storage.properties.joomlafile
                                                       searchElementsByTagName_helper() (in mod-
         method), 579
                                                                 ule translate.misc.ourdom), 393
```

±		(trans-	sentence_iter()	(translate.lang.ne.ne)	class
	late.convert.convert.Replacer method),		method), 376		
=	archreplacetemplate() (tra		sentence_iter()	(translate.lang.pa.pa	class
late.convert.convert.Replacer method), 240			method), 377	1 . 1 1	a D
segmentfile()	(in module	trans-	sentence_iter()(ta= 378	ranslate.lang.si.si class	method),
late.tools.pose		1 leter-161 e		uamalata lama at at alaaa	mathad)
SELF_CLOSING_TAG. attribute), 435		ı.nımıjue	sentence_iter()(ta= 379	ransiaie.iang.si.si ciass	тетоа),
sentence_iter()	(in module	tuana	sentence_iter()	(translate.lang.sv.sv	class
late.search.seg	•	trans-	method), 380	(transtate.tang.sv.sv	ciuss
sentence_iter()	(translate.lang.af.af	class	sentence_iter()	(translate.lang.ta.ta	class
<i>method</i> ), 353	(iransiaie.iang.aj.aj	ciuss	method), 381	(transtate.tang.ta.ta	ciuss
sentence_iter()	(translate.lang.am.am	class	sentence_iter()(t	ranslate lano te te class	method)
<i>method</i> ), 354	(iranstate.tang.am.am	ciass	382	anstate.tang.te.te etass	memou),
sentence_iter()	(translate.lang.ar.ar	class	sentence_iter()	(translate.lang.th.th	class
method), 355	(*		method), 382	(	
sentence_iter()	(translate.lang.bn.bn	class	sentence_iter()	(translate.lang.ug.ug	class
method), 355	`		method), 383	. 000	
sentence_iter()	(translate.lang.code_or	:code_or	sentence_iter()	(translate.lang.ur.ur	class
class method),	_		method), 384		
sentence_iter()	(translate.lang.common.	Common	sentence_iter()(t	ranslate.lang.vi.vi class	method),
class method),	359		385		
sentence_iter()	(translate.lang.de.de	class	sentence_iter()	(translate.lang.zh.zh	class
method), 363			method), 386		
sentence_iter()(	translate.lang.el.el class	method),	sentencecount()		(trans-
364			late.filters.chec	ks.CCLicenseChecker	method),
sentence_iter()	(translate.lang.es.es	class	272		
method), 365			sentencecount()		(trans-
<pre>sentence_iter() method), 366</pre>	(translate.lang.fa.fa	class	late.filters.chec 278	ks.DrupalChecker	method),
<pre>sentence_iter() (</pre>	translate.lang.fi.fi class	method),	sentencecount()		(trans-
366			late.filters.chec	ks.GnomeChecker	method),
<pre>sentence_iter() (</pre>	translate.lang.fr.fr class	method),	284		
367			sentencecount()		(trans-
sentence_iter()	(translate.lang.gu.gu	class		ks.IOSChecker method	), 290
method), 368			sentencecount()		(trans-
sentence_iter()	(translate.lang.he.he	class	late.filters.chec	ks.KdeChecker	method),
method), 369			295		
sentence_iter()	(translate.lang.hi.hi	class	sentencecount()		(trans-
method), 370				ks.L20nChecker	method),
sentence_iter()	(translate.lang.hy.hy	class	301		
method), 370			sentencecount()		(trans-
sentence_iter()	(translate.lang.ja.ja	class		ks.LibreOfficeChecker	method),
method), 371	(, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,	1	307		(1
sentence_iter()	(translate.lang.km.km	class	sentencecount()	ka MinimalChaskan	(trans-
method), 372	(tuanalata lana ku ku	alaaa		ks.MinimalChecker	method),
sentence_iter()	(translate.lang.kn.kn	class	313		(tuana
<pre>method), 373 sentence_iter()</pre>	(translate.lang.ko.ko	class	sentencecount()	ks.MozillaChecker	(trans- method),
method), 374		ciass	319	кѕ.могшаспескет	тетоа),
sentence_iter()	(translate.lang.ml.ml	class	sentencecount()		(trans-
method), 374		_	*	ks.OpenOfficeChecker	method),
<pre>sentence_iter()</pre>	(translate.lang.mr.mr	class	324		,
method), 375			sentencecount()		(trans-

late.filters.checks.ReducedChecker method),	serialize() (translate.storage.csvl10n.csvfile
330	method), 421
sentencecount() (trans- late.filters.checks.StandardChecker method),	serialize() (translate.storage.dtd.dtdfile method), 427
336	serialize() (translate.storage.html.htmlfile method),
sentencecount() (trans-	437
late.filters.checks.TermChecker method),	serialize() (translate.storage.html.POHTMLParser
343	method), 434
sentenceend (translate.lang.common.Common attribute), 359	serialize() (translate.storage.ical.icalfile method), 442
sentences() (in module translate.search.segment), 407	serialize() (translate.storage.ini.inifile method), 447
sentences () (translate.lang.af.af class method), 353	serialize() (translate.storage.jsonl10n.I18NextFile
sentences () (translate.lang.am.am class method),	method), 453
354	serialize() (translate.storage.jsonl10n.JsonFile
sentences () (translate.lang.ar.ar class method), 355	method), 458
sentences () (translate.lang.bn.bn class method), 355	serialize() (trans-
sentences() (translate.lang.code_or.code_or class	late.storage.jsonl10n.JsonNestedFile method),
method), 356	459
sentences() (translate.lang.common.Common class	serialize() (trans-
method), 359	late. storage. js on 110 n. Web Extension Js on File
sentences () (translate.lang.de.de class method), 363	method), 467
sentences () (translate.lang.el.el class method), 364	serialize() (translate.storage.lisa.LISAfile method),
sentences () (translate.lang.es.es class method), 365	472
sentences () (translate.lang.fa.fa class method), 366	serialize() (translate.storage.mo.mofile method),
sentences () (translate.lang.fi.fi class method), 367	478
sentences () (translate.lang.fr.fr class method), 367	serialize() (trans-
sentences () (translate.lang.gu.gu class method), 368	late.storage.mozilla_lang.LangStore method),
sentences () (translate.lang.he.he class method), 369	484
sentences () (translate.lang.hi.hi class method), 370	<pre>serialize() (translate.storage.omegat.OmegaTFile</pre>
sentences () (translate.lang.hy.hy class method), 370	method), 489
sentences () (translate.lang.ja.ja class method), 371	serialize() (trans-
sentences() (translate.lang.km.km class method),	late.storage.omegat.OmegaTFileTab method),
372	490
sentences () (translate.lang.kn.kn class method), 373	serialize() (translate.storage.oo.oofile method),
sentences () (translate.lang.ko.ko class method), 374	495
sentences () (translate.lang.ml.ml class method), 374	<pre>serialize() (translate.storage.php.phpfile method),</pre>
sentences () (translate.lang.mr.mr class method), 375	548
sentences () (translate.lang.ne.ne class method), 376	serialize() (translate.storage.pocommon.pofile
sentences () (translate.lang.pa.pa class method), 377	method), 554
sentences () (translate.lang.si.si class method), 378	serialize() (translate.storage.poxliff.PoXliffFile
sentences () (translate.lang.st.st class method), 379	method), 562
sentences () (translate.lang.sv.sv class method), 380	serialize() (translate.storage.properties.javafile
sentences () (translate.lang.ta.ta class method), 381	method), 576
sentences () (translate.lang.te.te class method), 382	serialize() (translate.storage.properties.javautf8file
sentences () (translate.lang.th.th class method), 382	method), 578
sentences () (translate.lang.ug.ug class method), 383	$\verb serialize()  (translate.storage.properties.joomla file$
sentences () (translate.lang.ur.ur class method), 384	method), 579
sentences () (translate.lang.vi.vi class method), 385	serialize() (translate.storage.properties.propfile
sentences () (translate.lang.zh.zh class method), 386	method), 581
serialize() (translate.storage.base.TranslationStore	serialize() (translate.storage.properties.stringsfile
method), 409	method), 586
serialize() (translate.storage.catkeys.CatkeysFile	serialize() (trans-
method), 416	late.storage.properties.stringsutf8file method),

588	set_timestring() (trans-
serialize() (translate.storage.pypo.pofile method), 591	late.storage.trados.TradosTxtDate method), 640
<pre>serialize() (translate.storage.qm.qmfile method),</pre>	set_timestring() (trans-
596	late. storage. word fast. Word fast Time  method),
serialize() (translate.storage.qph.QphFile method), 602	668 set_usage() (trans-
serialize() (translate.storage.rc.rcfile method), 607	late.convert.convert.ArchiveConvertOptionParser
serialize() (trans-	method), 236
late. storage. subtitles. Adv SubStation Alpha File	set_usage() (trans-
method), 614	late.convert.convert.ConvertOptionParser
serialize() (trans-	method), 239
$late. storage. subtitles. {\it MicroDVDFile}  method),$	set_usage() (trans-
615	late.convert.po2moz.MozConvertOptionParser
serialize() (translate.storage.subtitles.SubRipFile	method), 251
method), 617	set_usage() (trans-
serialize() (trans-	late.convert.po2tmx.TmxOptionParser
late.storage.subtitles.SubStationAlphaFile	method), 257
method), 619	set_usage() (trans-
serialize() (translate.storage.subtitles.SubtitleFile method), 620	late.convert.po2wordfast.WfOptionParser method), 261
<pre>serialize() (translate.storage.tbx.tbxfile method),</pre>	set_usage() (trans-
625	late.filters.pofilter.FilterOptionParser method),
serialize() (translate.storage.tiki.TikiStore method),	350
631	set_usage() (trans-
serialize() (translate.storage.tmx.tmxfile method), 636	late.misc.optrecurse.RecursiveOptionParser method), 392
serialize() (trans-	set_usage() (trans-
late.storage.trados.TradosTxtTmFile method), 645	late.tools.poconflicts.ConflictOptionParser method), 685
serialize() (translate.storage.ts2.tsfile method), 647	set_usage() (trans-
serialize() (translate.storage.txt.TxtFile method), 652	late.tools.pogrep.GrepOptionParser method), 689
<pre>serialize() (translate.storage.utx.UtxFile method),</pre>	set_usage() (trans-
657	late.tools.porestructure.SplitOptionParser
serialize() (trans-	method), 692
$late. storage. word fast. Word fast TMF ile\ method),$	set_usage() (trans-
668	late. tools. poterminology. Terminology Option Parser
serialize() (translate.storage.xliff.xlifffile method),	method), 696
674	setconfig() (trans-
SeriousFilterFailure, 332	late.filters.checks.CCLicenseChecker method),
set_field_analyzers() (trans-	272
	atabaseonfig() (translate.filters.checks.DrupalChecker
method), 399	method), 278
	setconfig() (translate.filters.checks.GnomeChecker
late.search.indexing.PyLuceneIndexer.PyLuceneI	
<pre>method), 402 set_field_analyzers() (trans-</pre>	setconfig() (translate.filters.checks.IOSChecker
late.search.indexing.XapianIndexer.XapianDatab	method), 290 pasetconfig() (translate.filters.checks.KdeChecker
method), 404	method), 296
<pre>set_time() (translate.storage.trados.TradosTxtDate</pre>	setconfig() (translate.filters.checks.L20nChecker method), 302
<pre>set_time() (translate.storage.wordfast.WordfastTime</pre>	setconfig() (trans-
method), 668	late.filters.checks.LibreOfficeChecker method),

307	482
setconfig() (trans- late.filters.checks.MinimalChecker method), 313	setcontext() (trans- late.storage.mozilla_lang.LangUnit method), 487
setconfig() (translate.filters.checks.MozillaChecker	setcontext() (trans-
method), 319	late.storage.omegat.OmegaTUnit method),
setconfig() (trans-	494
late.filters.checks.OpenOfficeChecker method),	setcontext() (translate.storage.php.phpunit
324	method), 551
setconfig() (trans-	setcontext() (translate.storage.pocommon.pounit
late.filters.checks.ReducedChecker method),	method), 557
330	setcontext() (translate.storage.poxliff.PoXliffUnit
setconfig() (trans-	method), 567
late.filters.checks.StandardChecker method),	setcontext() (translate.storage.properties.propunit
336	method), 584
setconfig() (trans-	setcontext() (translate.storage.pypo.pounit
late. filters. checks. Standard Unit Checker	method), 594
method), 339	setcontext() (translate.storage.qm.qmunit method),
setconfig() (translate.filters.checks.TermChecker	600
method), 343	setcontext() (translate.storage.qph.QphUnit
setconfig() (trans-	method), 605
late.filters.checks.TranslationChecker method),	setcontext() (translate.storage.rc.rcunit method),
346	610
setconfig() (translate.filters.checks.UnitChecker	
method), 346	late.storage.subtitles.SubtitleUnit method), 623
setcontext() (trans- late.storage.base.TranslationUnit method),	setcontext() (translate.storage.tbx.tbxunit method),
413	629
setcontext() (translate.storage.catkeys.CatkeysUnit	setcontext() (translate.storage.tiki.TikiUnit
method), 419	method), 634
setcontext() (translate.storage.csvl10n.csvunit	setcontext() (translate.storage.tmx.tmxunit
method), 424	method), 639
setcontext() (translate.storage.dtd.dtdunit method),	setcontext() (translate.storage.trados.TradosUnit
430	method), 643
setcontext() (translate.storage.html.htmlunit	setcontext() (translate.storage.ts2.tsunit method),
method), 440	650
setcontext() (translate.storage.ical.icalunit	<pre>setcontext() (translate.storage.txt.TxtUnit method),</pre>
method), 445	655
setcontext() (translate.storage.ini.iniunit method),	setcontext() (translate.storage.utx.UtxUnit
450	method), 661
setcontext() (trans-	setcontext() (trans-
late.storage.jsonl10n.I18NextUnit method),	late.storage.wordfast.WordfastUnit method),
456	671
setcontext() (trans- late.storage.jsonl10n.JsonNestedUnit method),	setcontext() (translate.storage.xliff.xliffunit
463	method), 678 setdefault() (translate.misc.dictutils.cidict
setcontext() (translate.storage.jsonl10n.JsonUnit	method), 386
method), 466	setdefault() (translate.storage.oo.unormalizechar
setcontext() (trans-	method), 496
late.storage.jsonl10n.WebExtensionJsonUnit	setdict() (translate.storage.catkeys.CatkeysUnit
method), 470	method), 419
setcontext() (translate.storage.lisa.LISAunit	setdict() (translate.storage.omegat.OmegaTUnit
method), 476	method), 494
setcontext() (translate.storage.mo.mounit method),	setdict() (translate.storage.utx.UtxUnit method),

661	method), 261
<pre>setdict() (translate.storage.wordfast.WordfastUnit</pre>	setformats() (trans-
method), 671	late.filters.pofilter.FilterOptionParser method),
seterrorleveloptions() (trans-	350
late.convert.convert.ArchiveConvertOptionParser	rsetformats() (trans-
method), 236	late.misc.optrecurse.RecursiveOptionParser
seterrorleveloptions() (trans-	method), 392
late.convert.convert.ConvertOptionParser	setformats() (trans-
method), 239	late.tools.poconflicts.ConflictOptionParser
seterrorleveloptions() (trans-	method), 685
late.convert.po2moz.MozConvertOptionParser	setformats() (trans-
method), 251	late.tools.pogrep.GrepOptionParser method),
seterrorleveloptions() (trans-	689
late.convert.po2tmx.TmxOptionParser	setformats() (trans-
method), 257	
	late.tools.porestructure.SplitOptionParser
seterrorleveloptions() (trans-	method), 692
late.convert.po2wordfast.WfOptionParser	setformats() (trans-
method), 261	late.tools.poterminology.TerminologyOptionParser
seterrorleveloptions() (trans-	method), 696
late.filters.pofilter.FilterOptionParser method),	setid() (translate.storage.base.TranslationUnit
350	method), 413
seterrorleveloptions() (trans-	setid() (translate.storage.catkeys.CatkeysUnit
late.misc.opt recurse. Recursive Option Parser	method), 419
method), 392	<pre>setid() (translate.storage.csvl10n.csvunit method),</pre>
seterrorleveloptions() (trans-	424
late. tools. poconflicts. Conflict Option Parser	setid() (translate.storage.dtd.dtdunit method), 431
method), 685	setid() (translate.storage.html.htmlunit method), 440
seterrorleveloptions() (trans-	setid() (translate.storage.ical.icalunit method), 445
late.tools.pogrep.GrepOptionParser method),	setid() (translate.storage.ini.iniunit method), 450
689	setid() (translate.storage.jsonl10n.I18NextUnit
seterrorleveloptions() (trans-	method), 456
late.tools.porestructure.SplitOptionParser	<pre>setid() (translate.storage.jsonl10n.JsonNestedUnit</pre>
method), 692	method), 463
seterrorleveloptions() (trans-	<pre>setid() (translate.storage.jsonl10n.JsonUnit method),</pre>
late.tools.poterminology.TerminologyOptionPars	
method), 696	setid() (translate.storage.jsonl10n.WebExtensionJsonUnit
setfilename() (translate.storage.poxliff.PoXliffFile	method), 470
method), 563	setid() (translate.storage.lisa.LISAunit method), 476
setfilename() (translate.storage.xliff.xlifffile	setid() (translate.storage.mo.mounit method), 482
method), 674	setid() (translate.storage.mozilla_lang.LangUnit
setformats() (trans-	method), 487
late.convert.convert.ArchiveConvertOptionParser	
method), 236	method), 494
setformats() (trans-	setid() (translate.storage.php.phpunit method), 552
late.convert.convert.ConvertOptionParser	setid() (translate.storage.pocommon.pounit method),
method), 239	557
setformats() (trans-	setid() (translate.storage.poxliff.PoXliffUnit method),
late.convert.po2moz.MozConvertOptionParser	567
method), 251	setid() (translate.storage.properties.propunit
setformats() (trans-	method), 584
late.convert.po2tmx.TmxOptionParser	setid() (translate.storage.pypo.pounit method), 594
method), 257	setid() (translate.storage.qm.qmunit method), 600
setformats() (trans-	setid() (translate.storage.qph.QphUnit method), 605
late.convert.po2wordfast.WfOptionParser	setid() (translate.storage.rc.rcunit method), 610

setid()	(translate.storage.subtitles.Subtitle Unit	method), 236	
1	method), 624	setpotoption()	(trans-
setid()	(translate.storage.tbx.tbxunit method), 629	late.convert.convert.ConvertOptionPars	er
setid()	(translate.storage.tiki.TikiUnit method), 634	method), 239	
setid()	(translate.storage.tmx.tmxunit method), 639	setpotoption()	(trans-
	(translate.storage.trados.TradosUnit method). 643	late.convert.po2moz.MozConvertOption method), 251	Parser
setid()	(translate.storage.ts2.tsunit method), 650	setpotoption()	(trans-
	(translate.storage.txt.TxtUnit method), 656 (translate.storage.utx.UtxUnit method), 661	late.convert.po2tmx.TmxOptionParser method), 257	
setid()	(translate.storage.wordfast.WordfastUnit	setpotoption()	(trans-
1	method), 672	late.convert.po2wordfast.WfOptionPars	er
setid()	(translate.storage.xliff.xliffunit method), 678	method), 262	
setmanp	ageoption() (trans-	setprogressoptions()	(trans-
	late.convert.convert.ArchiveConvertOptionPars method), 236	er late.convert.convert.ArchiveConvertOpt method), 236	ionParser
	ageoption() (trans-	setprogressoptions()	(trans-
	late.convert.convert.ConvertOptionParser method), 239	late.convert.convert.ConvertOptionPars method), 240	er
setmanp	ageoption() (trans-	setprogressoptions()	(trans-
	late.convert.po2moz.MozConvertOptionParser method), 251	late.convert.po2moz.MozConvertOption method), 251	Parser
setmanp	ageoption() (trans-	setprogressoptions()	(trans-
	late.convert.po2tmx.TmxOptionParser method), 257	late.convert.po2tmx.TmxOptionParser method), 257	
Ì	ageoption () (trans- late.convert.po2wordfast.WfOptionParser method), 261	setprogressoptions()  late.convert.po2wordfast.WfOptionPars  method), 262	(trans- er
setmanp	ageoption() (trans-	± 2 ± ···	(trans-
	late.filters.pofilter.FilterOptionParser method) 351	late.filters.pofilter.FilterOptionParser n 351	ıethod),
setmanp	ageoption() (trans-	setprogressoptions()	(trans-
	late.misc.optrecurse.RecursiveOptionParser method), 392	late.misc.optrecurse.RecursiveOptionPa method), 393	irser
Ì	ageoption () (trans- late.tools.poconflicts.ConflictOptionParser method), 685	setprogressoptions()  late.tools.poconflicts.ConflictOptionPar  method), 685	(trans- ser
setmanp	ageoption() (trans-	setprogressoptions()	(trans-
	late.tools.pogrep.GrepOptionParser method) 689	late.tools.pogrep.GrepOptionParser n 689	nethod),
setmanp	ageoption() (trans-	setprogressoptions()	(trans-
	late.tools.porestructure.SplitOptionParser method), 692	late.tools.porestructure.SplitOptionPars method), 693	er
	ageoption() (trans-	1 3 1	(trans-
	late.tools.poterminology.TerminologyOptionPa method), 696	rser late.tools.poterminology.TerminologyOp method), 696	otionParse
setpara	meters() (translate.search.match.matcher	setprojectstyle()	(trans-
1	method), 406	late.storage.base.TranslationStore n	nethod),
setpara	meters() (trans-	409	
	late.search.match.terminologymatcher	setprojectstyle()	(trans-
	method), 407		iethod),
	s () (translate.storage.oo.ooline method), 495	416	
setpoto		1 3 1 1	(trans-
l	$late.convert.convert.Archive {\it Convert Option Pars}$	er late.storage.csvl10n.csvfile method), 42	I

setprojectstyle() (translate.storage.dtd.dtdfile method), 427	late.storage.properties.propfile method), 581
setprojectstyle() (translate.storage.html.htmlfile	setprojectstyle() (trans-
<pre>method), 437 setprojectstyle() (trans-</pre>	late.storage.properties.stringsfile method), 586
late.storage.html.POHTMLParser method),	setprojectstyle() (trans-
434	late.storage.properties.stringsutf8file method),
setprojectstyle() (translate.storage.ical.icalfile method), 442	588 setprojectstyle() (translate.storage.pypo.pofile
setprojectstyle() (translate.storage.ini.inifile	method), 591
method), 447	setprojectstyle() (translate.storage.qm.qmfile
setprojectstyle() (trans-	method), 596
late.storage.jsonl10n.118NextFile method), 453	setprojectstyle() (translate.storage.qph.QphFile method), 602
setprojectstyle() (trans-	setprojectstyle() (translate.storage.rc.rcfile
late.storage.jsonl10n.JsonFile method), 458	method), 607
setprojectstyle() (trans-	setprojectstyle() (trans-
late.storage.jsonl10n.JsonNestedFile method), 459	late.storage.subtitles.AdvSubStationAlphaFile method), 614
setprojectstyle() (trans-	setprojectstyle() (trans-
late.storage.jsonl10n.WebExtensionJsonFile method), 467	late.storage.subtitles.MicroDVDFile method), 615
setprojectstyle() (translate.storage.lisa.LISAfile	setprojectstyle() (trans-
method), 472	late.storage.subtitles.SubRipFile method),
setprojectstyle() (translate.storage.mo.mofile	617
method), 479	setprojectstyle() (trans-
<pre>setprojectstyle() (trans- late.storage.mozilla_lang.LangStore method),</pre>	late.storage.subtitles.SubStationAlphaFile method), 619
484	setprojectstyle() (trans-
setprojectstyle() (trans-	late.storage.subtitles.SubtitleFile method),
late.storage.omegat.OmegaTFile method),	620
489	setprojectstyle() (translate.storage.tbx.tbxfile
setprojectstyle() (trans-	method), 625
late.storage.omegat.OmegaTFileTab method), 491	setprojectstyle() (translate.storage.tiki.TikiStore method), 631
<pre>setprojectstyle() (translate.storage.php.phpfile</pre>	setprojectstyle() (translate.storage.tmx.tmxfile method), 636
setprojectstyle() (trans-	setprojectstyle() (trans-
late.storage.pocommon.pofile method), 554	late. storage. trados. Trados Txt TmFile  method),
setprojectstyle() (trans-	645
late.storage.poheader.poheader method), 559	setprojectstyle() (translate.storage.ts2.tsfile method), 647
setprojectstyle() (trans- late.storage.poxliff.PoXliffFile method), 563	setprojectstyle() (translate.storage.txt.TxtFile
setprojectstyle() (trans-	method), 652 setprojectstyle() (translate.storage.utx.UtxFile
late.storage.properties.javafile method),	method), 658
576	setprojectstyle() (trans-
setprojectstyle() (trans-	late.storage.wordfast.WordfastTMFile method),
late.storage.properties.javautf8file method),	668
578	setprojectstyle() (translate.storage.xliff.xlifffile
setprojectstyle() (trans-	method), 674
late.storage.properties.joomlafile method),	setsourcelanguage() (trans-
580	late.storage.base.TranslationStore method),
setprojectstyle() (trans-	409

setsourcelanguage()	(trans-	580	
late. storage. catkeys. Catkeys File	method),	setsourcelanguage()	(trans-
416		late.storage.properties.propfile	method),
setsourcelanguage()	(trans-	581	
late.storage.csvl10n.csvfile method),		setsourcelanguage()	(trans-
<pre>setsourcelanguage() (translate.storage</pre>		late.storage.properties.stringsfile 586	method),
setsourcelanguage()		setsourcelanguage()	(trans-
late.storage.html.htmlfile method), 43		late.storage.properties.stringsutf8file	method),
setsourcelanguage()	(trans-	588	
late.storage.html.POHTMLParser	method),	setsourcelanguage()	(trans-
434		late.storage.pypo.pofile method), 591	C1
<pre>setsourcelanguage()     late.storage.ical.icalfile method), 442</pre>		setsourcelanguage() (translate.storage. method), 597	qm.qmfile
setsourcelanguage() (translate.storag	e.ini.inifile		(trans-
method), 447		late.storage.qph.QphFile method), 60	
setsourcelanguage()	(trans-	setsourcelanguage() (translate.storage	ze.rc.rcfile
late.storage.jsonl10n.I18NextFile	method),	method), 607	
453		setsourcelanguage()	(trans-
<pre>setsourcelanguage()     late.storage.jsonl10n.JsonFile metho</pre>	(trans- d), 458	late.storage.subtitles.AdvSubStationA method), 614	lphaFile
setsourcelanguage()	(trans-	setsourcelanguage()	(trans-
late.storage.jsonl10n.JsonNestedFile 459	method),	late.storage.subtitles.MicroDVDFile 615	method),
setsourcelanguage()	(trans-	setsourcelanguage()	(trans-
late.storage.jsonl10n.WebExtensionJ method), 467	sonFile	late.storage.subtitles.SubRipFile 617	method),
setsourcelanguage()	(trans-	setsourcelanguage()	(trans-
late.storage.lisa.LISAfile method), 47	72	late.storage.subtitles.SubStationAlpho	aFile
setsourcelanguage() (translate.storage	.mo.mofile	method), 619	
method), 479		setsourcelanguage()	(trans-
setsourcelanguage()	(trans-	late. storage. subtitles. Subtitle File	method),
late.storage.mozilla_lang.LangStore	method),	620	
484		setsourcelanguage() (translate.storage.	tbx.tbxfile
setsourcelanguage()	(trans-	method), 625	
late.storage.omegat.OmegaTFile	method),	setsourcelanguage()	(trans-
489		late.storage.tiki.TikiStore method), 63	
setsourcelanguage()		setsourcelanguage()	(trans-
late.storage.omegat.OmegaTFileTab	method),	late.storage.tmx.tmxfile method), 636	
491		setsourcelanguage()	(trans-
setsourcelanguage()	(trans-	late.storage.trados.TradosTxtTmFile	method),
late.storage.php.phpfile method), 549		645	. 2 . 61
setsourcelanguage()	(trans-	setsourcelanguage() (translate.storag	e.ts2.tsfile
late.storage.pocommon.pofile method		method), 647	(tuana
setsourcelanguage()  late.storage.poxliff.PoXliffFile metho	(trans-	setsourcelanguage()	(trans-
	a), 303 (trans-	late.storage.txt.TxtFile method), 652	(trans
<pre>setsourcelanguage()     late.storage.properties.javafile</pre>	method),	setsourcelanguage()  late.storage.utx.UtxFile method), 658	(trans-
576	memou),	setsourcelanguage()	(trans-
setsourcelanguage()	(trans-	late.storage.wordfast.WordfastTMFile	
late.storage.properties.javautf8file	method),	668	, memou),
578	memou),	setsourcelanguage()	(trans-
setsourcelanguage()	(trans-	late.storage.xliff.xlifffile method), 675	`
late.storage.properties.joomlafile	method),	setsuggestionstore()	(trans-
	, ,		•

late.filters.checks.CCLicenseChecker method), 272	settarget() (translate.storage.tbx.tbxunit method), 629
${\it late.filters.checks.DrupalChecker} \qquad {\it (trans-late.filters.checks.DrupalChecker} \qquad {\it method}),$	settarget() (translate.storage.tmx.tmxunit method), 639
278	settarget() (translate.storage.ts2.tsunit method),
setsuggestionstore() (trans-	650
late.filters.checks.GnomeChecker method), 284	settarget() (translate.storage.xliff.xliffunit method), 678
setsuggestionstore() (trans-	settargetlanguage() (trans-
late.filters.checks.IOSChecker method), 290	$late.storage.base.TranslationStore \qquad method),$
setsuggestionstore() (trans-	409
late.filters.checks.KdeChecker method), 296	settargetlanguage() (trans- late.storage.catkeys.CatkeysFile method),
setsuggestionstore() (trans-	416
late.filters.checks.L20nChecker method),	settargetlanguage() (trans-
302	late.storage.catkeys.CatkeysHeader method),
setsuggestionstore() (trans-	416
late.filters.checks.LibreOfficeChecker method),	settargetlanguage() (trans-
307	late.storage.csvl10n.csvfile method), 421
setsuggestionstore() (trans-	settargetlanguage() (translate.storage.dtd.dtdfile
late.filters.checks.MinimalChecker method),	method), 428
313	settargetlanguage() (trans-
setsuggestionstore() (trans-	late.storage.html.htmlfile method), 437
late.filters.checks.MozillaChecker method),	settargetlanguage() (trans-
319	late.storage.html.POHTMLParser method),
setsuggestionstore() (trans-	434
late.filters.checks.OpenOfficeChecker method), 324	settargetlanguage() (trans-
	late.storage.ical.icalfile method), 442 settargetlanguage() (translate.storage.ini.inifile
setsuggestionstore() (trans- late.filters.checks.ReducedChecker method),	method), 447
330	settargetlanguage() (trans-
setsuggestionstore() (trans-	late.storage.jsonl10n.118NextFile method),
late.filters.checks.StandardChecker method),	453
336	settargetlanguage() (trans-
setsuggestionstore() (trans-	late.storage.jsonl10n.JsonFile method), 458
late.filters.checks.StandardUnitChecker	settargetlanguage() (trans-
method), 339	late.storage.jsonl10n.JsonNestedFile method),
setsuggestionstore() (trans-	459
late.filters.checks.TeeChecker method), 339	settargetlanguage() (trans-
setsuggestionstore() (trans-	late. storage. js on l10n. Web Extension Js on File
late.filters.checks.TermChecker method),	method), 467
343	settargetlanguage() (trans-
setsuggestionstore() (trans-	late.storage.lisa.LISAfile method), 472
late.filters.checks.TranslationChecker method), 346	settargetlanguage() (translate.storage.mo.mofile method), 479
setsuggestionstore() (trans-	settargetlanguage() (trans-
late.filters.checks.UnitChecker method), 346	late.storage.mozilla_lang.LangStore method), 484
settarget() (translate.storage.lisa.LISAunit	settargetlanguage() (trans-
method), 476	late.storage.omegat.OmegaTFile method),
<pre>settarget() (translate.storage.poxliff.PoXliffUnit     method), 567</pre>	489 settargetlanguage() (trans-
settarget() (translate.storage.qph.QphUnit method), 605	late.storage.omegat.OmegaTFileTab method), 491

<pre>settargetlanguage()     late.storage.php.phpfile method), 549</pre>	(trans-	settargetlanguage()  late.storage.tmx.tmxfile method), 636	(trans-
settargetlanguage()	(trans-	settargetlanguage()	(trans-
late.storage.pocommon.pofile method		late.storage.trados.TradosTxtTmFile	*
settargetlanguage()	(trans-	645	,,
late.storage.poheader.poheader 559	method),	settargetlanguage() (translate.storage method), 647	.ts2.tsfile
settargetlanguage()	(trans-	settargetlanguage()	(trans-
late.storage.poxliff.PoXliffFile method	d), 563	late.storage.txt.TxtFile method), 653	`
settargetlanguage()	(trans-	settargetlanguage()	(trans-
late.storage.properties.javafile 576	method),	late.storage.utx.UtxFile method), 658	(4
	(tuana	settargetlanguage()	(trans-
settargetlanguage()	(trans-	late.storage.wordfast.WordfastTMFile	тетпоа),
late.storage.properties.javautf8file	method),	668	( )
578		settargetlanguage()	(trans-
settargetlanguage()	(trans-	late.storage.xliff.xlifffile method), 675	D 407
late.storage.properties.joomlafile	method),	settext() (translate.storage.oo.ooline metho	
580		settimestampoption()	(trans-
settargetlanguage()	(trans-	late.convert.convert.ArchiveConvertOp	otionParser
late.storage.properties.propfile	method),	method), 236	
581		settimestampoption()	(trans-
settargetlanguage()	(trans-	late.convert.convert.ConvertOptionPar	rser
late.storage.properties.stringsfile	method),	method), 240	
586		settimestampoption()	(trans-
settargetlanguage()	(trans-	late.convert.po2moz.MozConvertOptio	nParser
late.storage.properties.stringsutf8file	method),	method), 252	
588		settimestampoption()	(trans-
settargetlanguage()	(trans-	late.convert.po2tmx.TmxOptionParser	
late.storage.pypo.pofile method), 591		method), 257	
<pre>settargetlanguage() (translate.storage</pre>	.qm.qmfile	<pre>settimestampoption()     late.convert.po2wordfast.WfOptionPar</pre>	(trans-
settargetlanguage()	(trans-	method), 262	
late.storage.qph.QphFile method), 60		settypecomment() (translate.storage.pyp	oo.pounit
settargetlanguage() (translate.stora		method), 594	<i>T</i>
method), 607	g,	setXMLlang() (in module	trans-
settargetlanguage()	(trans-	late.misc.xml_helpers), 396	
late.storage.subtitles.AdvSubStationA		setXMLspace() (in module	trans-
method), 614	7	late.misc.xml_helpers), 396	
settargetlanguage()	(trans-	short() (translate.filters.checks.CCLicense	eChecker
late.storage.subtitles.MicroDVDFile	method),	method), 272	
615	,,,	short() (translate.filters.checks.Drupa	lChecker
settargetlanguage()	(trans-	method), 278	. C. r. c. r. c.
late.storage.subtitles.SubRipFile	method),	short() (translate.filters.checks.Gnome	eChecker
617	memou),	method), 284	<i>cricener</i>
settargetlanguage()	(trans-	short() (translate.filters.checks.IOSChecker	method)
late.storage.subtitles.SubStationAlph	•	290	memou),
method), 619	ar ne	short() (translate.filters.checks.KdeChecker	method)
settargetlanguage()	(trans-	296	memou),
late.storage.subtitles.SubtitleFile	method),	short() (translate.filters.checks.L20r	nChecker
620		method), 302	
<pre>settargetlanguage() (translate.storage     method), 626</pre>	.tbx.tbxfile	short() (translate.filters.checks.LibreOffice method), 307	eChecker
settargetlanguage()	(trans-	short() (translate.filters.checks.Minima	lChecker
late.storage.tiki.TikiStore method), 63	`	method), 313	

short(	) (translate.filters.checks.MozillaCheck method), 319	er simpleplurals() (trans late.filters.checks.CCLicenseChecker method)
short(	1 00	
short(	method), 324 ) (translate.filters.checks.ReducedCheck method), 330	simpleplurals() (trans er late.filters.checks.DrupalChecker method) 279
short(		er simpleplurals() (trans
-1	method), 336	late.filters.checks.GnomeChecker method)
SHOLL (	) (translate.filters.checks.TermChecker method 343	simpleplurals() (trans
should_	_output_store() (in module tran	
-1()	late.convert.convert), 240	simpleplurals() (trans
show()	(translate.misc.progressbar.DotsProgressB method), 394	296
show()	(translate.misc.progressbar.NoProgressB	
show()	method), 394 (translate.misc.progressbar.ProgressB	late.filters.checks.L20nChecker method) ar 302
()	method), 394	simpleplurals() (trans
	in translate.lang.si), 378	late.filters.checks.LibreOfficeChecker method)
simple		
	late.filters.checks.CCLicenseChecker method 273	late.filters.checks.MinimalChecker method)
simple	caps () (tran late.filters.checks.DrupalChecker method	
	279	late.filters.checks.MozillaChecker method)
simple		
	late.filters.checks.GnomeChecker method 284	late.filters.checks.OpenOfficeChecker method)
simple	caps () (translate.filters.checks.IOSCheck method), 290	er 325 simpleplurals() (trans
simple	,, , , , , , , , , , , , , , , , , , ,	
simple	caps () (translate.filters.checks.L20nCheck method), 302	er simpleplurals() (trans late.filters.checks.StandardChecker method)
simple		
	$late. filters. checks. Libre Of fice Checker\ method$	(trans
simple	307 caps() (tran	late.filters.checks.TermChecker method) s- 343
- 1	late.filters.checks.MinimalChecker method	
simple	313 caps() (tran	simplify_to_common() (in module trans
01	late.filters.checks.MozillaChecker method	
	319	late.filters.checks.CCLicenseChecker method)
simple		
	late.filters.checks.OpenOfficeChecker method 325	(trans), singlequoting() (trans) late.filters.checks.DrupalChecker method)
simple		
	late.filters.checks.ReducedChecker method 330	late.filters.checks.GnomeChecker method)
simple		
	late.filters.checks.StandardChecker method 336	late.filters.checks.IOSChecker method), 290
simple		
	method), 343	late.filters.checks.KdeChecker method)

296	SourceStoreClass (translate.convert.po2tiki.po2tiki
singlequoting() (trans-	attribute), 253
late.filters.checks.L20nChecker method),	SourceStoreClass (trans-
302	late.convert.po2yaml.po2yaml attribute),
singlequoting() (trans-	262
late.filters.checks.LibreOfficeChecker method),	SourceStoreClass (translate.convert.tiki2po.tiki2po
308	attribute), 266
singlequoting() (trans-	SourceStoreClass (translate.convert.txt2po.txt2po
late.filters.checks.MinimalChecker method),	attribute), 267
313	SourceStoreClass (trans-
singlequoting() (trans-	late.convert.yaml2po.yaml2po attribute),
late.filters.checks.MozillaChecker method),	268
319	spaceend() (in module translate.filters.decoration),
	347
singlequoting() (trans-	
late.filters.checks.OpenOfficeChecker method),	spacestart() (in module trans-
325	late.filters.decoration), 347
singlequoting() (trans-	specialchars (translate.lang.common.Common at-
late.filters.checks.ReducedChecker method),	tribute), 359
331	spellcheck() (trans-
singlequoting() (trans-	late.filters.checks.CCLicenseChecker method),
late.filters.checks.StandardChecker method),	273
336	spellcheck() (trans-
singlequoting() (trans-	late.filters.checks.DrupalChecker method),
late.filters.checks.TermChecker method),	279
344	spellcheck() (trans-
<pre>slam() (translate.misc.wStringIO.CatchStringOutput</pre>	late.filters.checks.GnomeChecker method). 285
source (translate.storage.dtd.dtdunit attribute), 431	spellcheck() (translate.filters.checks.IOSChecker
source (translate.storage.pypo.pounit attribute), 594	method), 291
source_wordcount() (trans-	spellcheck() (translate.filters.checks.KdeChecker
late.storage.statistics.Statistics method),	method), 296
611	spellcheck() (translate.filters.checks.L20nChecker
sourcefiles (trans-	method), 302
late.storage.bundleprojstore.BundleProjectStore	
attribute), 414	
	late.filters.checks.LibreOfficeChecker method),
sourcefiles (translate.storage.projstore.ProjectStore	308
attribute), 568	spellcheck() (trans-
sourcelen() (in module translate.search.match), 406	
SourceStoreClass (trans-	314
late.convert.ical2po.ical2po attribute), 242	spellcheck() (trans-
SourceStoreClass (translate.convert.ini2po.ini2po	late.filters.checks.MozillaChecker method),
attribute), 243	319
SourceStoreClass (trans-	spellcheck() (trans-
late.convert.mozlang2po.lang2po attribute),	late.filters.checks.OpenOfficeChecker method),
244	325
SourceStoreClass (trans-	spellcheck() (trans-
late.convert.php2po.php2po attribute), 246	late.filters.checks.ReducedChecker method),
SourceStoreClass (trans-	331
late.convert.po2ical.po2ical attribute), 247	spellcheck() (trans-
SourceStoreClass (translate.convert.po2ini.po2ini	late.filters.checks.StandardChecker method),
attribute), 247	337
SourceStoreClass (trans-	spellcheck() (translate.filters.checks.TermChecker
late.convert.po2mozlang.po2lang attribute),	method), 344
248	split() (translate.misc.multistring.multistring

method), 389	late.tools.poterminology.TerminologyOptionParse
${\tt splitext}$ () (translate.convert.convert.ArchiveConvertConvertConvert.	
method), 236	splitlines() (in module translate.storage.pypo),
<pre>splitext() (translate.convert.ConvertOptionPage</pre>	
method), 240	splitlines() (translate.misc.multistring.multistring
<pre>splitext() (translate.convert.po2moz.MozConvertOpti method), 252</pre>	ConParser method), 389 SplitOptionParser (class in trans-
$\verb splitext()  (translate.convert.po2tmx.TmxOptionParse$	r late.tools.porestructure), 690
method), 257	splittemplateext() (trans-
<pre>splitext() (translate.convert.po2wordfast.WfOptionPa method), 262</pre>	arser late.convert.convert.ArchiveConvertOptionParser method), 236
<pre>splitext() (translate.filters.pofilter.FilterOptionParser     method), 351</pre>	splittemplateext() (trans- late.convert.convert.ConvertOptionParser
splitext() (translate.misc.optrecurse.RecursiveOption	
method), 393	splittemplateext() (trans-
<pre>splitext() (translate.tools.poconflicts.ConflictOptionF</pre>	Parser late.convert.po2moz.MozConvertOptionParser method), 252
<pre>splitext() (translate.tools.pogrep.GrepOptionParser</pre>	
method), 689	late.convert.po2tmx.TmxOptionParser
$\verb splitext()  (translate.tools.porestructure.SplitOptionPolicy   Splite   SplitOptionPolicy   SplitOptio$	arser method), 257
method), 693	splittemplateext() (trans-
<pre>splitext() (translate.tools.poterminology.Terminology     method), 696</pre>	OptionParskate.convert.po2wordfast.WfOptionParser method), 262
	splittemplateext() (trans-
late.convert.convert.ArchiveConvertOptionParsemethod), 236	r late.filters.pofilter.FilterOptionParser method), 351
splitinputext() (trans-	splittemplateext() (trans-
late.convert.convert.ConvertOptionParser method), 240	late.misc.optrecurse.RecursiveOptionParser method), 393
splitinputext() (trans-	splittemplateext() (trans-
late.convert.po2moz.MozConvertOptionParser method), 252	late.tools.poconflicts.ConflictOptionParser method), 685
splitinputext() (trans-	splittemplateext() (trans-
late.convert.po2tmx.TmxOptionParser method), 257	late.tools.pogrep.GrepOptionParser method), 689
splitinputext() (trans-	splittemplateext() (trans-
late.convert.po2wordfast.WfOptionParser method), 262	late.tools.porestructure.SplitOptionParser method), 693
splitinputext() (trans-	splittemplateext() (trans-
late.filters.pofilter.FilterOptionParser method), 351	late.tools.poterminology.TerminologyOptionParse. method), 696
splitinputext() (trans-	st (class in translate.lang.st), 379
late.misc.optrecurse.RecursiveOptionParser method), 393	StandardChecker (class in translate.filters.checks), 332
splitinputext() (trans-	StandardUnitChecker (class in trans-
late.tools.poconflicts.ConflictOptionParser	late.filters.checks), 338
method), 685	startcaps() (trans-
splitinputext() (trans-	late.filters.checks.CCLicenseChecker method),
late.tools.pogrep.GrepOptionParser method), 689	273 startcaps() (translate.filters.checks.DrupalChecker
splitinputext() (trans-	method), 279
late.tools.porestructure.SplitOptionParser method), 693	startcaps() (translate.filters.checks.GnomeChecker method), 285
splitinputext() (trans-	startcaps() (translate.filters.checks.IOSChecker

method), 291	startpunc() (translate.filters.checks.TermChecker
startcaps() (translate.filters.checks.KdeChecker	method), 344
method), 296	startswith() (translate.misc.multistring.multistring
startcaps() (translate.filters.checks.L20nChecker	method), 389 startwhitespace() (trans-
method), 302 startcaps() (trans-	late.filters.checks.CCLicenseChecker method),
late.filters.checks.LibreOfficeChecker method),	273
308	startwhitespace() (trans-
startcaps() (trans-	late.filters.checks.DrupalChecker method),
late.filters.checks.MinimalChecker method),	279
314	startwhitespace() (trans-
startcaps() (translate.filters.checks.MozillaChecker method), 320	late.filters.checks.GnomeChecker method), 285
	startwhitespace() (trans-
late.filters.checks.OpenOfficeChecker method),	late.filters.checks.IOSChecker method), 291
325	startwhitespace() (trans-
startcaps() (trans- late.filters.checks.ReducedChecker method),	late.filters.checks.KdeChecker method), 297
331	startwhitespace() (trans-
startcaps() (trans-	late.filters.checks.L20nChecker method),
late.filters.checks.StandardChecker method),	303
337	startwhitespace() (trans-
startcaps() (translate.filters.checks.TermChecker method), 344	late.filters.checks.LibreOfficeChecker method), 308
startpunc() (trans-	startwhitespace() (trans-
late.filters.checks.CCLicenseChecker method), 273	late.filters.checks.MinimalChecker method), 314
$\verb startpunc()  \textit{(translate.filters.checks.DrupalChecker} \\$	
method), 279	late.filters.checks.MozillaChecker method),
startpunc() (translate.filters.checks.GnomeChecker	320
method), 285	startwhitespace() (trans-
startpunc() (translate.filters.checks.IOSChecker method), 291	late.filters.checks.OpenOfficeChecker method), 326
startpunc() (translate.filters.checks.KdeChecker	startwhitespace() (trans-
method), 297	late.filters.checks.ReducedChecker method),
startpunc() (translate.filters.checks.L20nChecker	331
method), 303	startwhitespace() (trans-
startpunc() (trans-	late.filters.checks.StandardChecker method), 337
late.filters.checks.LibreOfficeChecker method), 308	startwhitespace() (trans-
startpunc() (trans-	late.filters.checks.TermChecker method),
late.filters.checks.MinimalChecker method),	344
314	StateEnum (class in translate.storage.workflow), 672
startpunc() (translate.filters.checks.MozillaChecker method), 320	statefordb() (in module translate.storage.statsdb), 612
startpunc() (trans-	statemap (translate.storage.ts2.tsunit attribute), 651
late.filters.checks.OpenOfficeChecker method),	StateNotInWorkflowError,672
325	Statistics (class in translate.storage.statistics), 610
startpunc() (trans-	StatsCache (class in translate.storage.statsdb), 611
late.filters.checks.ReducedChecker method), 331	str2bool() (in module translate.tools.pomerge), 690 string_xpath (in module trans-
startpunc() (trans-	late.misc.xml_helpers), 396
late.filters.checks.StandardChecker method),	string_xpath_normalized (in module trans-
337	late.misc.xml_helpers), 396

StringElem late.storage	(class e.placeables.strelen	in n), 526	trans-	tabs()	(translate.filter 303	s.checks.L20nCheck	er method),
StringIO (class in stringsfile (class)	translate.misc.wSt	ringIO), 3		tabs()	(translate.f method), 308	ilters.checks.LibreO	fficeChecker
585				tabs()	(transla	te.filters.checks.Mini	imalChecker
stringsutf8fil	e (class	in	trans-		method), 314		
late.storage	e.properties), 586			tabs()	(transla	ate.filters.checks.Mo	zillaChecker
strip() (ta	ranslate.misc.multi	string.mul	tistring		method), 320		
method), 39	90			tabs()	(translate.f	ilters.checks.OpenO	ffice Checker
strip_html()(in		-			method), 326		
Sub (class in transla				tabs()	(transla	te.filters.checks.Redi	ıcedChecker
Sub (class in transla					method), 331		
sub (translate.storag tribute), 52	_	m.StringE	lem at-	tabs()	(translat method), 337	e.filters.checks.Stand	dardChecker
SubflowPlaceab	le (class	in	trans-	tabs()	(translate.filter	s.checks.TermCheck	er method),
_	e.placeables.interfa				344		
SubRipFile (class		e.subtitles	3), 616			translate.filters.ched	cks), 346
SubStationAlpha late.storage	aFile (class e.subtitles), 617	in	trans-	tagpro	perties() <i>late.filters.chec</i>	(in module ks), 347	trans-
SubtitleFile ( $c$	lass in translate.s	storage.su	btitles),	take_a	ction()		(trans-
619 SubtitleUnit (c	elass in translate.s	storage.su	btitles),		late.misc.optre	curse.ManPageOptic	on method),
621				target	(translate.stora	ge.dtd.dtdunit attrib	ute), 431
suggestions_in	_format		(trans-			ge.pypo.pounit attril	
late.storage	e.base.TranslationS	tore att	ribute),	target	(translate.stora	ge.txt.TxtUnit attribi	ute), 656
409				target	files		(trans-
suggestions_in_ late.storage	_format e.xliff.xlifffile attribi	ute), 675	(trans-		late.storage.but attribute), 414	ndleprojstore.Bundle	eProjectStore
summarize()(in n			ıt), 686	target	files (transla	te.storage.projstore.	ProjectStore
supported_file	s() (in $m$	ıodule	trans-		attribute), 568		-
late.storage	e.factory), 432			Target	StoreClass		(trans-
sv (class in translate	e.lang.sv), 379				late.convert.ica	ıl2po.ical2po attribu	te), 242
svn (class in transla	te.storage.versionc	ontrol.svn	), 665	Target	StoreClass	(translate.convert.in	ni2po.ini2po
swapcase() (ta	ranslate.misc.multi	string.mul	tistring		attribute), 243		
method), 39				Target	StoreClass		(trans-
<pre>swapdir() (in mod switchfile()</pre>					late.convert.mo	ozlang2po.lang2po	attribute),
method), 56	53			Target	StoreClass		(trans-
switchfile()(tre	anslate.storage.xlifj	f.xlifffile m	ethod),		late.convert.ph	p2po.php2po attribu	te), 246
675				Target	StoreClass		(trans-
Т					_	2ical.po2ical attribu	
1				Target		(translate.convert.p	o2ini.po2ini
ta (class in translate					attribute), 247		
TAB_UTF16 (in mod				Target	StoreClass	2 1 21	(trans-
tabs() (translemethod), 27	ate.filters.checks.C 74	CLicenseC	Checker		248	2mozlang.po2lang	attribute),
tabs() (tra method), 28	anslate.filters.check 80	s.Drupal <b>C</b>	Checker	Target	StoreClass ( attribute), 253	(translate.convert.po	2tiki.po2tiki
	anslate.filters.check	s.GnomeC	Checker	Target	StoreClass		(trans-
method), 28					late.convert.po.	2yaml.po2yaml	attribute),
tabs() (translate.)	filters.checks.IOSC	hecker m	ethod),	Target	263 StoreClass	(translate.convert.tik	:i2po.tiki2po
	filters.checks.KdeC	hecker m	ethod).	2	attribute), 266		
297				Target		(translate.convert.t.	xt2po.txt2po

attribute), 267	templateexists() (trans-
TargetStoreClass (trans-	late.misc.optrecurse.RecursiveOptionParser
late.convert.yaml2po.yaml2po attribute),	method), 393
268	templateexists() (trans-
<pre>TargetUnitClass (translate.convert.ical2po.ical2po</pre>	late. to ols. poconflicts. Conflict Option Parser
attribute), 242	method), 686
TargetUnitClass (translate.convert.ini2po.ini2po	templateexists() (trans-
attribute), 243	late.tools.pogrep.GrepOptionParser method),
TargetUnitClass (trans-	689
late.convert.mozlang2po.lang2po attribute),	templateexists() (trans-
244	late.tools.porestructure.SplitOptionParser
TargetUnitClass (translate.convert.php2po.php2po	method), 693
attribute), 246	templateexists() (trans-
TargetUnitClass (translate.convert.po2ical.po2ical	late.tools.poterminology.TerminologyOptionParser
attribute), 247	method), 696
TargetUnitClass (translate.convert.po2ini.po2ini	TermChecker (class in translate filters.checks), 339
attribute), 248	terminologymatcher (class in trans-
TargetUnitClass (trans-	late.search.match), 406
late.convert.po2mozlang.po2lang attribute), 248	TerminologyOptionParser (class in translate.tools.poterminology), 694
TargetUnitClass (translate.convert.po2tiki.po2tiki	TerminologyPlaceable (class in trans-
attribute), 253	late.storage.placeables.terminology), 528
TargetUnitClass (trans-	text (translate.storage.oo.ooline attribute), 495
late.convert.po2yaml.po2yaml attribute),	th (class in translate.lang.th), 382
263	tiki2po (class in translate.convert.tiki2po), 266
TargetUnitClass (translate.convert.tiki2po.tiki2po	TikiStore (class in translate.storage.tiki), 629
attribute), 266	TikiUnit (class in translate.storage.tiki), 631
TargetUnitClass (translate.convert.txt2po.txt2po attribute), 267	time (translate.storage.trados.TradosTxtDate attribute), 640
TargetUnitClass (trans-	time (translate.storage.wordfast.WordfastTime at-
late.convert.yaml2po.yaml2po attribute),	tribute), 668
268	timestring (translate.storage.trados.TradosTxtDate
tbxfile (class in translate.storage.tbx), 624	attribute), 640
tbxunit (class in translate.storage.tbx), 626	timestring (translate.storage.wordfast.WordfastTime
te (class in translate.lang.te), 381	attribute), 668
TeeChecker (class in translate.filters.checks), 339	title() (translate.misc.multistring.multistring
templateexists() (trans-	method), 390
	r TMServer (class in translate.services.tmserver), 407
method), 236	tmxfile (class in translate.storage.tmx), 634
templateexists() (trans- late.convert.convert.ConvertOptionParser	TmxOptionParser (class in trans- late.convert.po2tmx), 254
method), 240	tmxunit (class in translate.storage.tmx), 636
templateexists() (trans-	tr_lang() (in module translate.lang.data), 363
late.convert.po2moz.MozConvertOptionParser	TRADOS_TIMEFORMAT (in module trans-
method), 252	late.storage.trados), 640
templateexists() (trans-	TradosTxtDate (class in translate.storage.trados),
late.convert.po2tmx.TmxOptionParser	640
method), 257	TradosTxtTmFile (class in translate.storage.trados),
templateexists() (trans-	643
late.convert.po2wordfast.WfOptionParser	TradosUnit (class in translate.storage.trados), 640
method), 262	tranliterate_cyrillic() (in module trans-
templateexists() (trans-	late.lang.af), 353
late.filters.pofilter.FilterOptionParser method),	transaction() (in module translate.storage.statsdb),
351	612

${\tt transfiles} \ ({\it translate.storage.bundle projstore.Bundle Following}) \\$	
attribute), 414	translate() (translate.storage.placeables.base.Ex
transfiles (translate.storage.projstore.ProjectStore	method), 508
attribute), 568	translate() (translate.storage.placeables.base.G
TransitionError, 672	method), 505
Translatable (class in translate.storage.xml_extract.extract), 678	translate() (translate.storage.placeables.base.It method), 503
translate() (translate.misc.multistring.multistring method), 390	
	translate() (translate.storage.placeables.base.Sub method), 511
translate() (translate.storage.catkeys.CatkeysFile method), 416	
translate() (translate.storage.csvl10n.csvfile	translate() (trans-
method), 421	late.storage.placeables.general.AltAttrPlaceable
translate() (translate.storage.dtd.dtdfile method), 428	method), 513
	translate() (trans- late.storage.placeables.general.XMLEntityPlaceable
translate() (translate.storage.html.htmlfile method), 437	method), 515
${\tt translate()} \ ({\it translate.storage.html.POHTMLParser}$	
method), 434	late. storage. placeables. general. XMLT ag Placeable
<pre>translate() (translate.storage.ical.icalfile method),</pre>	method), 517
442	translate() (trans-
translate() (translate.storage.ini.inifile method), 447	late.storage.placeables.interfaces.BasePlaceable method), 518
<pre>translate() (translate.storage.jsonl10n.I18NextFile</pre>	translate() (trans-
method), 453	late. storage. placeables. interfaces. In visible Placeable
translate() (translate.storage.jsonl10n.JsonFile	method), 520
method), 458	translate() (trans-
translate() (trans-	late. storage. placeables. interfaces. Masking Placeable
$late. storage. js on 110 n. Js on Nested File \ method),$	method), 522
460	translate() (trans-
translate() (trans-	late. storage. placeables. interfaces. Replacement Placeable
late. storage. js on l10n. Web Extension Js on File	method), 524
method), 467	translate() (trans-
translate() (translate.storage.lisa.LISAfile method), 472	late.storage.placeables.interfaces.SubflowPlaceable method), 525
<pre>translate() (translate.storage.mo.mofile method),</pre>	translate() (trans-
479	late. storage. placeables. strelem. String Elem
translate() (trans-	method), 528
late.storage.mozilla_lang.LangStore method),	translate() (trans-
484	late. storage. placeables. terminology. Terminology Placeable
${\tt translate()}  \textit{(translate.storage.omegat.OmegaTFile}$	method), 530
method), 489	translate() (translate.storage.placeables.xliff.Bpt
translate() (trans-	method), 532
late.storage.omegat.OmegaTFileTab method), 491	translate() (translate.storage.placeables.xliff.Bx method), 536
translate() (translate.storage.php.phpfile method), 549	translate() (translate.storage.placeables.xliff.Ept method), 533
translate() (translate.storage.placeables.base.Bpt method), 499	translate() (translate.storage.placeables.xliff.Ex method), 538
translate() (translate.storage.placeables.base.Bx method), 507	
translate() (translate.storage.placeables.base.Ept	

method), 541	late.storage.trados.TradosTxtTmFile method),
translate() (translate.storage.placeables.xliff.Ph	645
method), 544	translate() (translate.storage.ts2.tsfile method), 647
translate() (translate.storage.placeables.xliff.Sub method), 543	translate() (translate.storage.txt.TxtFile method), 653
translate() (trans-	
late.storage.placeables.xliff.UnknownXML	translate() (translate.storage.utx.UtxFile method), 658
method), 546	translate() (trans-
translate() (translate.storage.placeables.xliff.X	late.storage.wordfast.WordfastTMFile method),
method), 535	668
translate() (translate.storage.pocommon.pofile method), 554	translate() (translate.storage.xliff.xlifffile method), 675
translate() (translate.storage.poxliff.PoXliffFile	translate.convert (module), 231, 232
method), 563	translate.convert.accesskey(module), 232
translate() (translate.storage.properties.javafile	translate.convert.convert (module), 233
method), 576	translate.convert.csv2po(module), 240
translate() (translate.storage.properties.javautf8file	translate.convert.csv2tbx(module),241
method), 578	translate.convert.dtd2po(module),241
translate() (translate.storage.properties.joomlafile	translate.convert.factory(module),242
method), 580	translate.convert.html2po(module),242
translate() (translate.storage.properties.propfile	translate.convert.ical2po(module),242
method), 581	translate.convert.ini2po(module),243
translate() (translate.storage.properties.stringsfile	translate.convert.json2po(module),243
method), 586	translate.convert.moz2po(module), 244
translate() (trans-	translate.convert.mozfunny2prop (module),
late.storage.properties.stringsutf8file method), 588	244 translate.convert.mozlang2po ( <i>module</i> ), 244
translate() (translate.storage.pypo.pofile method),	translate.convert.odf2xliff(module), 245
591	translate.convert.oo2po(module), 245
translate() (translate.storage.qm.qmfile method),	translate.convert.oo2xliff(module), 245
597	translate.convert.php2po(module), 245
translate() (translate.storage.qph.QphFile method),	translate.convert.po2csv(module), 246
602	translate.convert.po2dtd(module),246
translate() (translate.storage.rc.rcfile method), 607	translate.convert.po2html(module),246
translate() (trans-	translate.convert.po2ical(module),247
late.storage.subtitles.AdvSubStationAlphaFile	translate.convert.po2ini(module),247
method), 614	translate.convert.po2json(module), 248
translate() (trans-	translate.convert.po2moz(module), 249
late.storage.subtitles.MicroDVDFile method),	translate.convert.po2mozlang (module), 248
615	translate.convert.po2oo(module), 252
translate() (translate.storage.subtitles.SubRipFile	translate.convert.po2php(module), 252
method), 617	translate.convert.po2prop(module), 252
translate() (trans-	translate.convert.po2rc(module), 253
late.storage.subtitles.SubStationAlphaFile	translate.convert.po2resx(module), 253
method), 619 translate() (translate.storage.subtitles.SubtitleFile	translate.convert.po2sub(module), 253
method), 621	translate.convert.po2symb(module),253
translate() (translate.storage.tbx.tbxfile method),	translate.convert.po2tiki (module), 253 translate.convert.po2tmx (module), 254
626 (transiale.siorage.tox.toxytte method),	translate.convert.po2ts (module), 258
translate() (translate.storage.tiki.TikiStore method),	translate.convert.po2ts (module), 258
631	translate.convert.po2web2py (module), 258
translate() (translate.storage.tmx.tmxfile method),	translate.convert.po2wordfast (module),
636	258
translate() (trans-	translate.convert.po2xliff(module), 262

```
translate.convert.po2yaml (module), 262
                                             translate.lang.pa (module), 376
translate.convert.pot2po (module), 263
                                             translate.lang.poedit (module), 377
translate.convert.prop2mozfunny (module),
                                             translate.lang.si(module), 378
                                             translate.lang.st (module), 379
translate.convert.prop2po (module), 264
                                             translate.lang.sv (module), 379
translate.convert.rc2po (module), 264
                                             translate.lang.ta(module), 380
translate.convert.resx2po (module), 265
                                             translate.lang.te(module), 381
                                             translate.lang.team (module), 381
translate.convert.sub2po (module), 265
translate.convert.symb2po (module), 266
                                             translate.lang.th (module), 382
translate.convert.tiki2po (module), 266
                                             translate.lang.ug (module), 383
translate.convert.ts2po (module), 266
                                             translate.lang.ur (module), 383
translate.convert.txt2po (module), 266
                                             translate.lang.vi (module), 384
translate.convert.web2py2po (module), 267
                                             translate.lang.zh (module), 385
                                             translate.misc (module), 232, 386
translate.convert.xliff2odf (module), 267
translate.convert.xliff2oo (module), 267
                                             translate.misc.dictutils (module), 386
translate.convert.xliff2po (module), 268
                                             translate.misc.file_discovery
                                                                                  (module),
                                                     387
translate.convert.yaml2po (module), 268
translate.filters (module), 231, 268
                                             translate.misc.lru (module), 387
translate.filters.autocorrect
                                             translate.misc.multistring (module), 387
                                    (module),
       268
                                             translate.misc.optrecurse (module), 390
translate.filters.checks (module), 269
                                             translate.misc.ourdom (module), 393
translate.filters.decoration (module), 347
                                             translate.misc.progressbar (module), 394
translate.filters.helpers (module), 348
                                             translate.misc.quote (module), 394
translate.filters.pofilter (module), 348
                                             translate.misc.wsgi (module), 395
translate.filters.prefilters (module), 351
                                             translate.misc.wStringIO (module), 395
translate.filters.spelling (module), 352
                                             translate.misc.xml helpers (module), 396
translate.lang (module), 231, 352
                                             translate.search (module), 232, 397
translate.lang.af (module), 352
                                             translate.search.indexing (module), 397
translate.lang.am (module), 353
                                             translate.search.indexing.CommonIndexer
translate.lang.ar (module), 354
                                                     (module), 397
translate.lang.bn (module), 355
                                             translate.search.indexing.PyLuceneIndexer
translate.lang.code_or(module), 356
                                                     (module), 400
translate.lang.common (module), 356
                                             translate.search.indexing.XapianIndexer
translate.lang.data(module), 360
                                                     (module), 402
translate.lang.de (module), 363
                                             translate.search.lshtein (module), 405
translate.lang.el (module), 363
                                             translate.search.match (module), 405
translate.lang.es (module), 364
                                             translate.search.segment (module), 407
translate.lang.fa (module), 365
                                             translate.search.terminology (module), 407
translate.lang.factory (module), 365
                                             translate.services (module), 232, 407
translate.lang.fi(module), 366
                                             translate.services.tmserver (module), 407
                                             translate.storage (module), 232, 408
translate.lang.fr (module), 367
translate.lang.gu(module),368
                                             translate.storage._factory_classes(mod-
translate.lang.he (module), 368
                                                     ule), 431
translate.lang.hi (module), 369
                                             translate.storage.base (module), 408
translate.lang.hy (module), 370
                                             translate.storage.benchmark (module), 413
translate.lang.identify (module), 371
                                             translate.storage.bundleprojstore (mod-
translate.lang.ja (module), 371
                                                     ule), 413
translate.lang.km (module), 372
                                             translate.storage.catkeys (module), 414
translate.lang.kn (module), 372
                                             translate.storage.csvl10n (module), 420
translate.lang.ko (module), 373
                                             translate.storage.directory (module), 425
translate.lang.ml (module), 374
                                             translate.storage.dtd(module), 425
translate.lang.mr (module), 375
                                             translate.storage.factory (module), 431
                                             translate.storage.html(module),432
translate.lang.ne (module), 375
translate.lang.ngram (module), 376
                                             translate.storage.ical (module), 441
```

```
translate.storage.ini (module), 446
                                             translate.storage.versioncontrol
translate.storage.jsonl10n (module), 451
                                                    ule), 661
translate.storage.lisa(module), 471
                                             translate.storage.versioncontrol.bzr
translate.storage.mo (module), 476
                                                    (module), 663
translate.storage.mozilla lang
                                   (module),
                                             translate.storage.versioncontrol.cvs
       482
                                                    (module), 663
translate.storage.odf io (module), 487
                                             translate.storage.versioncontrol.darcs
translate.storage.odf_shared(module),487
                                                    (module), 664
                                             translate.storage.versioncontrol.git
translate.storage.omegat (module), 487
                                                    (module), 664
translate.storage.oo(module), 494
                                             translate.storage.versioncontrol.hg
translate.storage.php (module), 546
translate.storage.placeables (module), 497
                                                    (module), 664
translate.storage.placeables.base (mod-
                                             translate.storage.versioncontrol.svn
       ule), 497
                                                    (module), 665
                                             translate.storage.wordfast (module), 665
translate.storage.placeables.general
       (module), 512
                                             translate.storage.workflow(module), 672
translate.storage.placeables.interfaces translate.storage.xliff (module), 672
       (module), 517
                                             translate.storage.xml extract
                                                                                 (module),
translate.storage.placeables.lisa (mod-
                                                    678
       ule), 525
                                             translate.storage.xml extract.extract
translate.storage.placeables.parse(mod-
                                                    (module), 678
                                             translate.storage.xml_extract.generate
                                                    (module), 679
translate.storage.placeables.strelem
       (module), 526
                                             translate.storage.xml extract.misc(mod-
translate.storage.placeables.terminology
                                                    ule), 680
       (module), 528
                                             translate.storage.xml_extract.unit_tree
translate.storage.placeables.xliff(mod-
                                                    (module), 680
       ule), 530
                                             translate.storage.xml_extract.xpath_breadcrumb
translate.storage.po (module), 559
                                                    (module), 680
translate.storage.pocommon (module), 552
                                             translate.storage.xml_name (module), 681
translate.storage.poheader (module), 558
                                             translate.storage.zip (module), 681
translate.storage.poparser (module), 559
                                             translate.tools (module), 232, 682
                                             translate.tools.build_tmdb (module), 682
translate.storage.poxliff (module), 560
                                             translate.tools.phppo2pypo(module),682
translate.storage.project (module), 567
translate.storage.projstore (module), 568
                                             translate.tools.poclean (module), 682
translate.storage.properties (module), 569
                                             translate.tools.pocompile (module), 682
translate.storage.pypo (module), 588
                                             translate.tools.poconflicts (module), 683
translate.storage.qm(module),595
                                             translate.tools.pocount (module), 686
translate.storage.qph (module), 600
                                             translate.tools.podebug (module), 686
translate.storage.rc(module),606
                                             translate.tools.pogrep(module), 686
translate.storage.statistics(module), 610
                                             translate.tools.pomerge (module), 690
translate.storage.statsdb (module), 611
                                             translate.tools.porestructure
                                                                                 (module),
translate.storage.subtitles (module), 612
translate.storage.symbian (module), 624
                                             translate.tools.posegment (module), 693
translate.storage.tbx (module), 624
                                             translate.tools.poswap (module), 693
translate.storage.tiki(module), 629
                                             translate.tools.poterminology
                                                                                 (module),
translate.storage.tmdb(module), 634
                                             translate.tools.pretranslate (module), 697
translate.storage.tmx (module), 634
translate.storage.trados (module), 639
                                             translate.tools.pydiff (module), 697
translate.storage.ts (module), 651
                                             translate.tools.pypo2phppo (module), 698
translate.storage.ts2 (module), 645
                                             TranslateBenchmarker
                                                                       (class
                                                                               in
                                                                                    trans-
translate.storage.txt (module), 651
                                                    late.storage.benchmark), 413
translate.storage.utx (module), 656
                                             translated_unitcount()
                                                                                   (trans-
                                                    late.storage.statistics.Statistics
                                                                                 method),
```

611	unchanged() (trans-
translated_units() (trans-	late.filters.checks.StandardChecker method),
late.storage.statistics.Statistics method),	337
611	unchanged() (translate.filters.checks.TermChecker
translated_wordcount() (trans-	method), 344
late.storage.statistics.Statistics method),	unescape() (in module translate.storage.pypo), 594
611	unescape() (in module translate.storage.trados), 640
TranslationChecker (class in trans-	unescape_help_text() (in module trans-
late.filters.checks), 345	late.storage.oo), 496
translations (trans-	unescape_text() (in module translate.storage.oo),
late.storage.placeables.terminology.Terminology	
attribute), 530	unified_diff() (translate.tools.pydiff.FileDiffer
TranslationStore (class in translate.storage.base),	method), 698
408	unit2dict() (in module translate.search.match), 407
TranslationUnit (class in translate.storage.base), 410	unit_iter() (translate.storage.base.TranslationStore method), 410
tsfile (class in translate.storage.ts2), 645	<pre>unit_iter() (translate.storage.base.TranslationUnit</pre>
tsunit (class in translate.storage.ts2), 647	method), 413
txt2po (class in translate.convert.txt2po), 267	<pre>unit_iter() (translate.storage.catkeys.CatkeysFile</pre>
TxtFile (class in translate.storage.txt), 651	method), 416
TxtUnit (class in translate.storage.txt), 653	<pre>unit_iter() (translate.storage.catkeys.CatkeysUnit</pre>
tzstring() (in module translate.storage.poheader),	method), 419
559	unit_iter() (translate.storage.csvl10n.csvfile
U	method), 421
	unit_iter() (translate.storage.csvl10n.csvunit
ug (class in translate.lang.ug), 383	method), 425
unchanged() (trans-	unit_iter() (translate.storage.directory.Directory
late.filters.checks.CCLicenseChecker method),	method), 425
274	<pre>unit_iter() (translate.storage.dtd.dtdfile method),</pre>
unchanged() (translate.filters.checks.DrupalChecker	428
method), 280	<pre>unit_iter() (translate.storage.dtd.dtdunit method),</pre>
unchanged() (translate.filters.checks.GnomeChecker	431
method), 286	unit_iter() (translate.storage.html.htmlfile method),
unchanged() (translate.filters.checks.IOSChecker	437 (translate storage html htmlunit
method), 291	unit_iter() (translate.storage.html.htmlunit method), 440
unchanged() (translate.filters.checks.KdeChecker	unit_iter() (translate.storage.html.POHTMLParser
method), 297 unchanged() (translate.filters.checks.L20nChecker	method), 434
unchanged() (translate.filters.checks.L20nChecker method), 303	unit_iter() (translate.storage.ical.icalfile method),
unchanged() (trans-	443
late.filters.checks.LibreOfficeChecker method),	unit_iter() (translate.storage.ical.icalunit method),
308	446
unchanged() (trans-	unit_iter() (translate.storage.ini.inifile method),
late.filters.checks.MinimalChecker method),	447
314	<pre>unit_iter() (translate.storage.ini.iniunit method),</pre>
unchanged() (translate.filters.checks.MozillaChecker	450
method), 320	<pre>unit_iter() (translate.storage.jsonl10n.118NextFile</pre>
unchanged() (trans-	method), 453
late.filters.checks.OpenOfficeChecker method),	unit_iter() (translate.storage.jsonl10n.I18NextUnit
326	method), 456
unchanged() (trans-	<pre>unit_iter() (translate.storage.jsonl10n.JsonFile</pre>
late.filters.checks.ReducedChecker method),	method), 458
331	unit_iter() (trans-
	late.storage.jsonl10n.JsonNestedFile method),

460	method), 584
unit_iter() (trans-	<pre>unit_iter() (translate.storage.properties.stringsfile</pre>
$late. storage. js on 110 n. Js on Nested Unit\ method),$	method), 586
463	unit_iter() (trans-
<pre>unit_iter() (translate.storage.jsonl10n.JsonUnit</pre>	late.storage.properties.stringsutf8file method),
method), 466	588
unit_iter() (trans-	<pre>unit_iter() (translate.storage.pypo.pofile method),</pre>
late.storage.jsonl10n.WebExtensionJsonFile	591
method), 468	<pre>unit_iter() (translate.storage.pypo.pounit method),</pre>
unit_iter() (trans-	594
late.storage.jsonl10n.WebExtensionJsonUnit	<pre>unit_iter() (translate.storage.qm.qmfile method),</pre>
method), 471	597
unit_iter() (translate.storage.lisa.LISAfile method), 473	unit_iter() (translate.storage.qm.qmunit method), 600
	unit_iter() (translate.storage.qph.QphFile method),
method), 476	602
unit_iter() (translate.storage.mo.mofile method),	unit_iter() (translate.storage.qph.QphUnit
479	method), 605
<pre>unit_iter() (translate.storage.mo.mounit method),</pre>	unit_iter() (translate.storage.rc.rcfile method), 607
482	unit_iter() (translate.storage.rc.rcunit method), 610
unit_iter() (trans-	unit_iter() (trans-
late.storage.mozilla_lang.LangStore method),	late.storage.subtitles.AdvSubStationAlphaFile
484	method), 614
<pre>unit_iter() (trans-</pre>	unit_iter() (trans-
late.storage.mozilla_lang.LangUnit method),	$late. storage. subtitles. {\it MicroDVDFile}  method),$
487	616
<pre>unit_iter() (translate.storage.omegat.OmegaTFile</pre>	<pre>unit_iter() (translate.storage.subtitles.SubRipFile</pre>
method), 489	method), 617
unit_iter() (trans-	unit_iter() (trans-
late.storage.omegat.OmegaTFileTab method),	late.storage.subtitles.SubStationAlphaFile
491	method), 619
unit_iter() (translate.storage.omegat.OmegaTUnit	
method), 494	method), 621
unit_iter() (translate.storage.php.phpfile method), 549	method), 624
<pre>unit_iter() (translate.storage.php.phpunit method),</pre>	
552	626
<pre>unit_iter() (translate.storage.pocommon.pofile</pre>	
method), 554	629
<pre>unit_iter() (translate.storage.pocommon.pounit</pre>	<pre>unit_iter() (translate.storage.tiki.TikiStore method),</pre>
method), 557	631
<pre>unit_iter() (translate.storage.poxliff.PoXliffFile</pre>	<pre>unit_iter() (translate.storage.tiki.TikiUnit method),</pre>
method), 563	634
<pre>unit_iter() (translate.storage.poxliff.PoXliffUnit</pre>	<pre>unit_iter() (translate.storage.tmx.tmxfile method),</pre>
method), 567	636
<pre>unit_iter() (translate.storage.properties.javafile</pre>	<pre>unit_iter() (translate.storage.tmx.tmxunit method),</pre>
method), 576	639
unit_iter() (translate.storage.properties.javautf8file	unit_iter() (trans-
method), 578	late.storage.trados.TradosTxtTmFile method),
unit_iter() (translate.storage.properties.joomlafile	645
method), 580	unit_iter() (translate.storage.trados.TradosUnit
unit_iter() (translate.storage.properties.propfile	<pre>method), 643 unit_iter() (translate.storage.ts2.tsfile method), 647</pre>
<pre>method), 581 unit_iter() (translate.storage.properties.propunit</pre>	unit_iter() (translate.storage.ts2.tspile method), 04/ unit_iter() (translate.storage.ts2.tsunit method),
unite_tet() (nansiaie.siorage.properties.propunii	unite_tet() (nansiate.siorage.is2.isunti method),

651	UnitClass (translate.storage.php.phpfile attribute),
unit_iter() (translate.storage.txt.TxtFile method),	547
653	UnitClass (translate.storage.pocommon.pofile at-
<pre>unit_iter() (translate.storage.txt.TxtUnit method),</pre>	tribute), 552
656	UnitClass (translate.storage.poxliff.PoXliffFile at-
<pre>unit_iter() (translate.storage.utx.UtxFile method),</pre>	tribute), 560
658	UnitClass (translate.storage.properties.javafile attribute), 575
<pre>unit_iter() (translate.storage.utx.UtxUnit method), 661</pre>	UnitClass (translate.storage.properties.javautf8file
unit_iter() (trans-	attribute), 576
late.storage.wordfast.WordfastTMFile method),	UnitClass (translate.storage.properties.joomlafile at-
668	tribute), 578
unit_iter() (trans-	UnitClass (translate.storage.properties.propfile at-
late.storage.wordfast.WordfastUnit method),	tribute), 580
672	UnitClass (translate.storage.properties.stringsfile at-
unit_iter() (translate.storage.xliff.xlifffile method), 675	tribute), 585 UnitClass (translate.storage.properties.stringsutf8file
unit_iter() (translate.storage.xliff.xliffunit method),	attribute), 586
678	UnitClass (translate.storage.pypo.pofile attribute),
<pre>unit_iter() (translate.storage.zip.ZIPFile method),</pre>	588
681	UnitClass (translate.storage.qm.qmfile attribute), 595
UnitChecker (class in translate.filters.checks), 346	UnitClass (translate.storage.qph.QphFile attribute),
UnitClass (translate.storage.base.TranslationStore at-	600
tribute), 408 UnitClass (translate.storage.catkeys.CatkeysFile at-	UnitClass (translate.storage.rc.rcfile attribute), 606 UnitClass (translate.storage.subtitles.AdvSubStationAlphaFile
tribute), 415	attribute), 612
UnitClass (translate.storage.csvl10n.csvfile attribute),	UnitClass (translate.storage.subtitles.MicroDVDFile
420	attribute), 614
UnitClass (translate.storage.dtd.dtdfile attribute), 426	UnitClass (translate.storage.subtitles.SubRipFile at-
UnitClass (translate.storage.html.htmlfile attribute),	tribute), 616
435 UnitClass (translate.storage.html.POHTMLParser at-	UnitClass (translate.storage.subtitles.SubStationAlphaFile attribute), 617
tribute), 432	UnitClass (translate.storage.subtitles.SubtitleFile at-
UnitClass (translate.storage.ical.icalfile attribute),	tribute), 619
441	UnitClass (translate.storage.tbx.tbxfile attribute), 624
UnitClass (translate.storage.ini.inifile attribute), 446	UnitClass (translate.storage.tiki.TikiStore attribute),
UnitClass (translate.storage.jsonl10n.118NextFile at-	630
tribute), 451	UnitClass (translate.storage.tmx.tmxfile attribute),
UnitClass (translate.storage.jsonl10n.JsonFile at- tribute), 456	634
UnitClass (translate.storage.jsonl10n.JsonNestedFile	UnitClass (translate.storage.trados.TradosTxtTmFile attribute), 643
attribute), 458	UnitClass (translate.storage.ts2.tsfile attribute), 645
	on Filat Class (translate.storage.txt.TxtFile attribute), 651
attribute), 466	UnitClass (translate.storage.utx.UtxFile attribute),
${\tt UnitClass} \ \ ({\it translate.storage.lisa.LISAfile} \ \ {\it attribute}),$	656
471	UnitClass (translate.storage.wordfast.WordfastTMFile
UnitClass (translate.storage.mo.mofile attribute), 476	attribute), 666
UnitClass (translate.storage.mozilla_lang.LangStore attribute), 482	UnitClass (translate.storage.xliff.xlifffile attribute), 672
UnitClass (translate.storage.omegat.OmegaTFile at-	UnitMixer (class in translate.convert.accesskey), 232
tribute), 487	unitstats() (translate.storage.statsdb.StatsCache
UnitClass (translate.storage.omegat.OmegaTFileTab	method), 612
attribute), 489	UnknownExtensionError, 242
UnitClass (translate.storage.oo.oofile attribute), 495	UnknownXML (class in trans-

late.storage.placeables.xliff), 545	late.storage.statistics.Statistics method),	
unormalizechar (class in translate.storage.oo), 496	611	
unquote_plus() (in module trans	1 " "	
late.storage.pocommon), 557	update() (translate.filters.checks.CheckerConfig	
unquotefromandroid() (in module trans		
late.storage.dtd), 431	update() (translate.misc.dictutils.cidict method), 386	
unquotefromdtd() (in module trans	1 "	
late.storage.dtd), 431	method), 496	
UnsupportedConversionError,242	update() (translate.storage.versioncontrol.bzr.bzr	
untranslated() (trans		
late.filters.checks.CCLicenseChecker method) 274	), update() (translate.storage.versioncontrol.cvs.cvs method), 663	
untranslated() (trans		
late.filters.checks.DrupalChecker method		
280	update() (translate.storage.versioncontrol.GenericRevisionControlSys	ste
untranslated() (trans	method), 662	
late.filters.checks.GnomeChecker method) 286	), update() (translate.storage.versioncontrol.git.git method), 664	
untranslated() (translate.filters.checks.IOSChecke		
method), 291	method), 665	
untranslated() (trans	1	
late.filters.checks.KdeChecker method)		
297	update_file() (trans-	
untranslated() (trans		
late.filters.checks.L20nChecker method)		
303	<pre>update_file() (translate.storage.project.Project</pre>	
untranslated() (trans	· · · · · · · · · · · · · · · · · · ·	
late.filters.checks.LibreOfficeChecker method		
309	late.storage.projstore.ProjectStore method),	
untranslated() (trans		
late.filters.checks.MinimalChecker method) 314	), updatecontributor() ( <i>translate.storage.mo.mofile method</i> ), 479	
untranslated() (trans	- updatecontributor() (trans-	
late.filters.checks.MozillaChecker method)	late.storage.pocommon.pofile method), 554	
320	updatecontributor() (trans-	
untranslated() (trans	0 1 1	
late.filters.checks.OpenOfficeChecker method		
326	updatecontributor() (trans-	
untranslated() (trans	0 1 33 33	
late.filters.checks.ReducedChecker method		
331	late.storage.pypo.pofile method), 591	
untranslated() (trans	± · · · · · · · · · · · · · · · · · · ·	
late.filters.checks.StandardChecker method	, , , , , , , , , , , , , , , , , , , ,	
337	updateheader() (translate.storage.mo.mofile	
untranslated() (trans		
late.filters.checks.TermChecker method) 344	), updateheader() (translate.storage.pocommon.pofile method), 554	
untranslated_unitcount() (trans	- updateheader() (trans-	
late.storage.statistics.Statistics method	), late.storage.poheader.poheader method), 559	
untranslated_units() (trans		
late.storage.statistics.Statistics method		
611	updateheader() (translate.storage.pypo.pofile	
untranslated_wordcount() (trans		

updateheaderplural() (trans-	V
late.storage.mo.mofile method), 479	<pre>valid_fieldnames() (in module trans-</pre>
updateheaderplural() (trans-	late.storage.csvl10n), 425
late.storage.pocommon.pofile method), 555	validaccel (translate.lang.common.Common at-
updateheaderplural() (trans-	tribute), 359
late.storage.poheader.poheader method), 559	validchars() (trans-
	late.filters.checks.CCLicenseChecker method),
updateheaderplural() (trans- late.storage.poxliff.PoXliffFile method), 563	274
updateheaderplural() (trans-	validchars() (trans-
late.storage.pypo.pofile method), 591	late.filters.checks.DrupalChecker method),
updatetargetlanguage() (trans-	280
late.filters.checks.CheckerConfig method),	validchars () (trans-
275	late.filters.checks.GnomeChecker method), 286
updatevalidchars() (trans-	validchars() (translate.filters.checks.IOSChecker
late.filters.checks.CheckerConfig method),	method), 291
275	validchars() (translate.filters.checks.KdeChecker
upper() (translate.misc.multistring.multistring	method), 297
method), 390	validchars() (translate.filters.checks.L20nChecker
ur (class in translate.lang.ur), 383	method), 303
urls() (translate.filters.checks.CCLicenseChecker	validchars() (trans-
method), 274	late.filters.checks.LibreOfficeChecker method),
urls() (translate.filters.checks.DrupalChecker	309
method), 280	validchars() (trans-
urls() (translate.filters.checks.GnomeChecker	late.filters.checks.MinimalChecker method),
method), 286	314
urls() (translate.filters.checks.IOSChecker method),	validchars() (trans-
291	late.filters.checks.MozillaChecker method),
urls() (translate.filters.checks.KdeChecker method),	320
297	validchars() (trans-
urls() (translate.filters.checks.L20nChecker method),	$late. filters. checks. Open Of fice Checker\ method),$
303	326
urls () (translate.filters.checks.LibreOfficeChecker	validchars() (trans-
method), 309	late.filters.checks.ReducedChecker method),
urls () (translate.filters.checks.MinimalChecker method), 314	332
urls() (translate.filters.checks.MozillaChecker	validchars() (trans-
method), 320	late.filters.checks.StandardChecker method),
urls() (translate.filters.checks.OpenOfficeChecker	337
method), 326	validchars () (translate.filters.checks.TermChecker method), 345
urls() (translate.filters.checks.ReducedChecker	<i>"</i>
method), 332	validdoublewords (trans- late.lang.common.Common attribute), 359
urls() (translate.filters.checks.StandardChecker	validxml() (translate.filters.checks.LibreOfficeChecke
method), 337	method), 309
urls() (translate.filters.checks.TermChecker method),	value_strip() (translate.storage.properties.Dialect
344	class method), 570
<pre>usable() (translate.search.match.matcher method),</pre>	value_strip() (trans-
406	late.storage.properties.DialectFlex class
usable() (translate.search.match.terminologymatcher	method), 571
method), $407$	value_strip() (trans-
UtxDialect (class in translate.storage.utx), 656	late.storage.properties.DialectGaia class
UtxFile (class in translate.storage.utx), 656	method), 571
UtxHeader (class in translate.storage.utx), 658	value_strip() (trans-
UtxUnit (class in translate.storage.utx), 658	late.storage.properties.DialectJava class

method), 572	variables() (translate.filters.checks.TermChecker
<pre>value_strip() (trans-</pre>	method), 345
late.storage.properties.DialectJavaUtf8 class method), 572	varname() (in module translate.filters.prefilters), 352 varnone() (in module translate.filters.prefilters), 352
value_strip() (trans-	verifyoptions() (in module trans-
late.storage.properties.DialectJoomla class	late.convert.oo2po), 245
method), 572	verifyoptions() (in module trans-
<pre>value_strip() (trans-</pre>	late.convert.oo2xliff), 245
late.storage.properties.DialectMozilla class	verifyoptions() (trans-
method), 573	late.convert.convert.ArchiveConvertOptionParser
<pre>value_strip() (trans-</pre>	method), 237
late.storage.properties.DialectSkype class	verifyoptions() (trans-
method), 573	late.convert.convert.ConvertOptionParser
<pre>value_strip() (trans-</pre>	method), 240
late.storage.properties.DialectStrings class	verifyoptions() (trans-
method), 574	late.convert.po2moz.MozConvertOptionParser
<pre>value_strip() (trans-</pre>	method), 252
${\it late.storage.properties.} Dialect Strings Utf 8$	verifyoptions() (trans-
class method), 574	late.convert.po2tmx.TmxOptionParser
valuerefs() (translate.misc.lru.LRUCachingDict	method), 257
method), 387	verifyoptions() (trans-
values() (translate.misc.dictutils.cidict method), 386	late.convert.po2wordfast.WfOptionParser
values() (translate.storage.oo.unormalizechar	method), 262
method), 496	vi (class in translate.lang.vi), 384
variables() (trans-	viewitems() (translate.misc.dictutils.cidict method),
late.filters.checks.CCLicenseChecker method),	386
274	viewitems() (translate.storage.oo.unormalizechar
variables() (translate.filters.checks.DrupalChecker	method), 496
method), 280	viewkeys() (translate.misc.dictutils.cidict method),
variables() (translate.filters.checks.GnomeChecker	386
method), 286	viewkeys () (translate.storage.oo.unormalizechar
variables() (translate.filters.checks.IOSChecker	method), 496 viewvalues() (translate.misc.dictutils.cidict
method), 292	•
variables() (translate.filters.checks.KdeChecker method), 297	method), 386
variables() (translate.filters.checks.L20nChecker	viewvalues() (translate.storage.oo.unormalizechar method), 496
method), 303	memou), 490
variables() (trans-	W
late.filters.checks.LibreOfficeChecker method),	
309	warning()(translate.convert.convert.ArchiveConvertOptionParser method), 237
variables() (trans-	warning() (translate.convert.convert.ConvertOptionParser
late.filters.checks.MinimalChecker method),	method), 240
315	warning()(translate.convert.po2moz.MozConvertOptionParser
variables() (translate.filters.checks.MozillaChecker	method), 252
method), 320	warning()(translate.convert.po2tmx.TmxOptionParser
variables() (trans-	method), 257
late.filters.checks.OpenOfficeChecker method),	warning() (translate.convert.po2wordfast.WfOptionParser
326	method), 262
variables() (trans-	warning() (translate.filters.pofilter.FilterOptionParser
late.filters.checks.ReducedChecker method),	method), 351
332	warning() (translate.misc.optrecurse.RecursiveOptionParser
variables() (trans-	method), 393
late.filters.checks.StandardChecker method),	warning()(translate.tools.poconflicts.ConflictOptionParser
338	method), 686

warning() (translate.tools.pogrep.GrepOptionParser method), 689	word_iter() (translate.lang.sv.sv class method), 380 word_iter() (translate.lang.ta.ta class method), 381
warning() (translate.tools.porestructure.SplitOptionPar method), 693	
warning() (translate.tools.poterminology.TerminologyC	
method), 697	word_iter() (translate.lang.ur.ur class method), 384
WebExtensionJsonFile (class in trans-	word_iter() (translate.lang.vi.vi class method), 385
late.storage.jsonl10n), 466	word_iter() (translate.lang.zh.zh class method), 386
WebExtensionJsonUnit (class in trans-	wordcount() (translate.storage.statistics.Statistics
late.storage.jsonl10n), 468	method), 611
WF_ESCAPE_MAP (in module trans-	WordfastDialect (class in trans-
late.storage.wordfast), 666	late.storage.wordfast), 666
WF_FIELDNAMES (in module trans-	WordfastHeader (class in trans-
late.storage.wordfast), 666	late.storage.wordfast), 666
WF_FIELDNAMES_HEADER (in module trans-	WordfastTime (class in translate.storage.wordfast),
late.storage.wordfast), 666	668
WF_FIELDNAMES_HEADER_DEFAULTS (in module	WordfastTMFile (class in trans-
translate.storage.wordfast), 666	late.storage.wordfast), 666
WF_TIMEFORMAT (in module trans-	WordfastUnit (class in translate.storage.wordfast),
late.storage.wordfast), 666	669
WfOptionParser (class in trans-	words () (in module translate.search.segment), 407
late.convert.po2wordfast), 258	words () (translate.lang.af.af class method), 353
<pre>word_iter() (in module translate.search.segment),</pre>	words () (translate.lang.am.am class method), 354
407	words () (translate.lang.ar.ar class method), 355
<pre>word_iter() (translate.lang.af.af class method), 353</pre>	words () (translate.lang.bn.bn class method), 355
<pre>word_iter() (translate.lang.am.am class method),</pre>	words() (translate.lang.code_or.code_or class method), 356
<pre>word_iter() (translate.lang.ar.ar class method), 355</pre>	words() (translate.lang.common.Common class
<pre>word_iter() (translate.lang.bn.bn class method), 355</pre>	method), 359
<pre>word_iter() (translate.lang.code_or.code_or class</pre>	words () (translate.lang.de.de class method), 363
<pre>word_iter() (translate.lang.code_or.code_or class</pre>	words () (translate.lang.de.de class method), 363 words () (translate.lang.el.el class method), 364
<pre>method), 356 word_iter() (translate.lang.common.Common class     method), 359</pre>	words () (translate.lang.el.el class method), 364 words () (translate.lang.es.es class method), 365 words () (translate.lang.fa.fa class method), 366
<pre>method), 356 word_iter() (translate.lang.common.Common class</pre>	words () (translate.lang.el.el class method), 364 words () (translate.lang.es.es class method), 365 words () (translate.lang.fa.fa class method), 366 words () (translate.lang.fi.fi class method), 367
<pre>method), 356 word_iter() (translate.lang.common.Common class</pre>	words () (translate.lang.el.el class method), 364 words () (translate.lang.es.es class method), 365 words () (translate.lang.fa.fa class method), 366 words () (translate.lang.fi.fi class method), 367 words () (translate.lang.fr.fr class method), 367
<pre>method), 356 word_iter() (translate.lang.common.Common class     method), 359 word_iter() (translate.lang.de.de class method), 363 word_iter() (translate.lang.el.el class method), 364 word_iter() (translate.lang.es.es class method), 365</pre>	words () (translate.lang.el.el class method), 364 words () (translate.lang.es.es class method), 365 words () (translate.lang.fa.fa class method), 366 words () (translate.lang.fi.fi class method), 367 words () (translate.lang.fr.fr class method), 367 words () (translate.lang.gu.gu class method), 368
<pre>method), 356 word_iter() (translate.lang.common.Common class</pre>	words () (translate.lang.el.el class method), 364 words () (translate.lang.es.es class method), 365 words () (translate.lang.fa.fa class method), 366 words () (translate.lang.fi.fi class method), 367 words () (translate.lang.fr.fr class method), 367 words () (translate.lang.gu.gu class method), 368 words () (translate.lang.he.he class method), 369
<pre>method), 356 word_iter() (translate.lang.common.Common class</pre>	words () (translate.lang.el.el class method), 364 words () (translate.lang.es.es class method), 365 words () (translate.lang.fa.fa class method), 366 words () (translate.lang.fi.fi class method), 367 words () (translate.lang.fr.fr class method), 367 words () (translate.lang.gu.gu class method), 368 words () (translate.lang.he.he class method), 369 words () (translate.lang.hi.hi class method), 370
method), 356 word_iter() (translate.lang.common.Common class method), 359 word_iter() (translate.lang.de.de class method), 363 word_iter() (translate.lang.el.el class method), 364 word_iter() (translate.lang.es.es class method), 365 word_iter() (translate.lang.fa.fa class method), 366 word_iter() (translate.lang.fi.fi class method), 367 word_iter() (translate.lang.fr.fr class method), 367	words () (translate.lang.el.el class method), 364 words () (translate.lang.es.es class method), 365 words () (translate.lang.fa.fa class method), 366 words () (translate.lang.fi.fi class method), 367 words () (translate.lang.fr.fr class method), 367 words () (translate.lang.gu.gu class method), 368 words () (translate.lang.he.he class method), 369 words () (translate.lang.hi.hi class method), 370 words () (translate.lang.hy.hy class method), 371
method), 356 word_iter() (translate.lang.common.Common class method), 359 word_iter() (translate.lang.de.de class method), 363 word_iter() (translate.lang.el.el class method), 364 word_iter() (translate.lang.es.es class method), 365 word_iter() (translate.lang.fa.fa class method), 366 word_iter() (translate.lang.fi.fi class method), 367 word_iter() (translate.lang.fr.fr class method), 367 word_iter() (translate.lang.gu.gu.gu class method), 368	words () (translate.lang.el.el class method), 364 words () (translate.lang.es.es class method), 365 words () (translate.lang.fa.fa class method), 366 words () (translate.lang.fi.fi class method), 367 words () (translate.lang.fr.fr class method), 367 words () (translate.lang.gu.gu class method), 368 words () (translate.lang.he.he class method), 369 words () (translate.lang.hi.hi class method), 370 words () (translate.lang.hy.hy class method), 371 words () (translate.lang.ja.ja class method), 371
method), 356  word_iter() (translate.lang.common.Common class method), 359  word_iter() (translate.lang.de.de class method), 363  word_iter() (translate.lang.el.el class method), 364  word_iter() (translate.lang.es.es class method), 365  word_iter() (translate.lang.fa.fa class method), 366  word_iter() (translate.lang.fi.fi class method), 367  word_iter() (translate.lang.fr.fr class method), 367  word_iter() (translate.lang.gu.gu class method), 368  word_iter() (translate.lang.he.he class method), 369	words () (translate.lang.el.el class method), 364 words () (translate.lang.es.es class method), 365 words () (translate.lang.fa.fa class method), 366 words () (translate.lang.fi.fi class method), 367 words () (translate.lang.fr.fr class method), 367 words () (translate.lang.gu.gu class method), 368 words () (translate.lang.he.he class method), 369 words () (translate.lang.hi.hi class method), 370 words () (translate.lang.hy.hy class method), 371 words () (translate.lang.ja.ja class method), 371 words () (translate.lang.km.km class method), 372
method), 356  word_iter() (translate.lang.common.Common class method), 359  word_iter() (translate.lang.de.de class method), 363  word_iter() (translate.lang.el.el class method), 364  word_iter() (translate.lang.es.es class method), 365  word_iter() (translate.lang.fa.fa class method), 366  word_iter() (translate.lang.fi.fi class method), 367  word_iter() (translate.lang.fr.fr class method), 367  word_iter() (translate.lang.gu.gu class method), 368  word_iter() (translate.lang.he.he class method), 369  word_iter() (translate.lang.hi.hi class method), 370	words () (translate.lang.el.el class method), 364 words () (translate.lang.es.es class method), 365 words () (translate.lang.fa.fa class method), 366 words () (translate.lang.fi.fi class method), 367 words () (translate.lang.fr.fr class method), 367 words () (translate.lang.gu.gu class method), 368 words () (translate.lang.he.he class method), 369 words () (translate.lang.hi.hi class method), 370 words () (translate.lang.hy.hy class method), 371 words () (translate.lang.ja.ja class method), 371 words () (translate.lang.km.km class method), 372 words () (translate.lang.kn.kn class method), 373
method), 356  word_iter() (translate.lang.common.Common class method), 359  word_iter() (translate.lang.de.de class method), 363  word_iter() (translate.lang.el.el class method), 364  word_iter() (translate.lang.es.es class method), 365  word_iter() (translate.lang.fa.fa class method), 366  word_iter() (translate.lang.fi.fi class method), 367  word_iter() (translate.lang.fr.fr class method), 367  word_iter() (translate.lang.gu.gu class method), 368  word_iter() (translate.lang.he.he class method), 369  word_iter() (translate.lang.hi.hi class method), 370  word_iter() (translate.lang.hy.hy class method), 371	words () (translate.lang.el.el class method), 364 words () (translate.lang.es.es class method), 365 words () (translate.lang.fa.fa class method), 366 words () (translate.lang.fi.fi class method), 367 words () (translate.lang.fr.fr class method), 367 words () (translate.lang.gu.gu class method), 368 words () (translate.lang.he.he class method), 369 words () (translate.lang.hi.hi class method), 370 words () (translate.lang.hy.hy class method), 371 words () (translate.lang.ja.ja class method), 371 words () (translate.lang.km.km class method), 372 words () (translate.lang.kn.kn class method), 373 words () (translate.lang.ko.ko class method), 374
method), 356  word_iter() (translate.lang.common.Common class method), 359  word_iter() (translate.lang.de.de class method), 363  word_iter() (translate.lang.el.el class method), 364  word_iter() (translate.lang.es.es class method), 365  word_iter() (translate.lang.fa.fa class method), 366  word_iter() (translate.lang.fi.fi class method), 367  word_iter() (translate.lang.fr.fr class method), 367  word_iter() (translate.lang.gu.gu class method), 368  word_iter() (translate.lang.he.he class method), 369  word_iter() (translate.lang.hi.hi class method), 370  word_iter() (translate.lang.hy.hy class method), 371  word_iter() (translate.lang.ja.ja class method), 371	words () (translate.lang.el.el class method), 364 words () (translate.lang.es.es class method), 365 words () (translate.lang.fa.fa class method), 366 words () (translate.lang.fi.fi class method), 367 words () (translate.lang.fr.fr class method), 367 words () (translate.lang.gu.gu class method), 368 words () (translate.lang.he.he class method), 369 words () (translate.lang.hi.hi class method), 370 words () (translate.lang.hy.hy class method), 371 words () (translate.lang.ja.ja class method), 371 words () (translate.lang.km.km class method), 372 words () (translate.lang.kn.kn class method), 373 words () (translate.lang.ko.ko class method), 374 words () (translate.lang.ml.ml class method), 375
method), 356  word_iter() (translate.lang.common.Common class method), 359  word_iter() (translate.lang.de.de class method), 363  word_iter() (translate.lang.el.el class method), 364  word_iter() (translate.lang.es.es class method), 365  word_iter() (translate.lang.fa.fa class method), 366  word_iter() (translate.lang.fi.fi class method), 367  word_iter() (translate.lang.fr.fr class method), 367  word_iter() (translate.lang.gu.gu class method), 368  word_iter() (translate.lang.he.he class method), 369  word_iter() (translate.lang.hi.hi class method), 370  word_iter() (translate.lang.hy.hy class method), 371  word_iter() (translate.lang.ja.ja class method), 371  word_iter() (translate.lang.km.km class method),	words () (translate.lang.el.el class method), 364 words () (translate.lang.es.es class method), 365 words () (translate.lang.fa.fa class method), 366 words () (translate.lang.fi.fi class method), 367 words () (translate.lang.fr.fr class method), 367 words () (translate.lang.gu.gu class method), 368 words () (translate.lang.he.he class method), 369 words () (translate.lang.hi.hi class method), 370 words () (translate.lang.hy.hy class method), 371 words () (translate.lang.ja.ja class method), 371 words () (translate.lang.km.km class method), 372 words () (translate.lang.kn.kn class method), 373 words () (translate.lang.ko.ko class method), 374 words () (translate.lang.ml.ml class method), 375 words () (translate.lang.mr.mr class method), 375
method), 356  word_iter() (translate.lang.common.Common class method), 359  word_iter() (translate.lang.de.de class method), 363  word_iter() (translate.lang.el.el class method), 364  word_iter() (translate.lang.es.es class method), 365  word_iter() (translate.lang.fa.fa class method), 366  word_iter() (translate.lang.fi.fi class method), 367  word_iter() (translate.lang.fr.fr class method), 367  word_iter() (translate.lang.gu.gu class method), 368  word_iter() (translate.lang.he.he class method), 369  word_iter() (translate.lang.hi.hi class method), 370  word_iter() (translate.lang.hy.hy class method), 371  word_iter() (translate.lang.ja.ja class method), 371  word_iter() (translate.lang.km.km class method), 371	words () (translate.lang.el.el class method), 364 words () (translate.lang.es.es class method), 365 words () (translate.lang.fa.fa class method), 366 words () (translate.lang.fi.fi class method), 367 words () (translate.lang.fr.fr class method), 367 words () (translate.lang.gu.gu class method), 368 words () (translate.lang.he.he class method), 369 words () (translate.lang.hi.hi class method), 370 words () (translate.lang.hy.hy class method), 371 words () (translate.lang.ja.ja class method), 371 words () (translate.lang.km.km class method), 372 words () (translate.lang.kn.kn class method), 373 words () (translate.lang.ko.ko class method), 374 words () (translate.lang.ml.ml class method), 375 words () (translate.lang.mr.mr class method), 375 words () (translate.lang.mr.mr class method), 376
method), 356  word_iter() (translate.lang.common.Common class method), 359  word_iter() (translate.lang.de.de class method), 363  word_iter() (translate.lang.el.el class method), 364  word_iter() (translate.lang.es.es class method), 365  word_iter() (translate.lang.fa.fa class method), 366  word_iter() (translate.lang.fi.fi class method), 367  word_iter() (translate.lang.fr.fr class method), 367  word_iter() (translate.lang.gu.gu class method), 368  word_iter() (translate.lang.he.he class method), 369  word_iter() (translate.lang.hi.hi class method), 370  word_iter() (translate.lang.hy.hy class method), 371  word_iter() (translate.lang.ja.ja class method), 371  word_iter() (translate.lang.km.km class method), 372  word_iter() (translate.lang.kn.kn class method), 373	words () (translate.lang.el.el class method), 364 words () (translate.lang.es.es class method), 365 words () (translate.lang.fa.fa class method), 366 words () (translate.lang.fi.fi class method), 367 words () (translate.lang.fr.fr class method), 367 words () (translate.lang.gu.gu class method), 368 words () (translate.lang.he.he class method), 369 words () (translate.lang.hi.hi class method), 370 words () (translate.lang.hy.hy class method), 371 words () (translate.lang.ja.ja class method), 371 words () (translate.lang.km.km class method), 372 words () (translate.lang.kn.kn class method), 373 words () (translate.lang.ml.ml class method), 374 words () (translate.lang.ml.ml class method), 375 words () (translate.lang.mr.mr class method), 376 words () (translate.lang.ne.ne class method), 376 words () (translate.lang.ne.ne class method), 376 words () (translate.lang.pa.pa class method), 377
method), 356  word_iter() (translate.lang.common.Common class method), 359  word_iter() (translate.lang.de.de class method), 363  word_iter() (translate.lang.el.el class method), 364  word_iter() (translate.lang.fe.fe class method), 365  word_iter() (translate.lang.fi.fe class method), 367  word_iter() (translate.lang.fi.fe class method), 367  word_iter() (translate.lang.gu.gu class method), 368  word_iter() (translate.lang.gu.gu class method), 369  word_iter() (translate.lang.he.he class method), 370  word_iter() (translate.lang.hy.hy class method), 371  word_iter() (translate.lang.ja.ja class method), 371  word_iter() (translate.lang.km.km class method), 372  word_iter() (translate.lang.kn.kn class method), 373  word_iter() (translate.lang.kn.kn class method), 373  word_iter() (translate.lang.kn.kn class method), 374	words () (translate.lang.el.el class method), 364 words () (translate.lang.es.es class method), 365 words () (translate.lang.fa.fa class method), 366 words () (translate.lang.fi.fi class method), 367 words () (translate.lang.fr.fr class method), 367 words () (translate.lang.gu.gu class method), 368 words () (translate.lang.he.he class method), 369 words () (translate.lang.hi.hi class method), 370 words () (translate.lang.hy.hy class method), 371 words () (translate.lang.ja.ja class method), 371 words () (translate.lang.km.km class method), 372 words () (translate.lang.kn.kn class method), 373 words () (translate.lang.ml.ml class method), 374 words () (translate.lang.ml.ml class method), 375 words () (translate.lang.mr.mr class method), 376 words () (translate.lang.ne.ne class method), 376 words () (translate.lang.pa.pa class method), 377 words () (translate.lang.si.si class method), 379
method), 356  word_iter() (translate.lang.common.Common class method), 359  word_iter() (translate.lang.de.de class method), 363  word_iter() (translate.lang.el.el class method), 364  word_iter() (translate.lang.es.es class method), 365  word_iter() (translate.lang.fa.fa class method), 366  word_iter() (translate.lang.fi.fi class method), 367  word_iter() (translate.lang.fr.fr class method), 367  word_iter() (translate.lang.gu.gu class method), 368  word_iter() (translate.lang.he.he class method), 369  word_iter() (translate.lang.hi.hi class method), 370  word_iter() (translate.lang.hy.hy class method), 371  word_iter() (translate.lang.km.km class method), 371  word_iter() (translate.lang.km.km class method), 372  word_iter() (translate.lang.kn.kn class method), 373  word_iter() (translate.lang.ko.ko class method), 374  word_iter() (translate.lang.ml.ml class method), 375	words () (translate.lang.el.el class method), 364 words () (translate.lang.es.es class method), 365 words () (translate.lang.fa.fa class method), 366 words () (translate.lang.fi.fi class method), 367 words () (translate.lang.fr.fr class method), 367 words () (translate.lang.gu.gu class method), 368 words () (translate.lang.he.he class method), 369 words () (translate.lang.hi.hi class method), 370 words () (translate.lang.hy.hy class method), 371 words () (translate.lang.ja.ja class method), 371 words () (translate.lang.km.km class method), 372 words () (translate.lang.kn.kn class method), 373 words () (translate.lang.ml.ml class method), 374 words () (translate.lang.ml.ml class method), 375 words () (translate.lang.mr.mr class method), 375 words () (translate.lang.ne.ne class method), 376 words () (translate.lang.pa.pa class method), 379 words () (translate.lang.si.si class method), 379 words () (translate.lang.si.si class method), 379
method), 356  word_iter() (translate.lang.common.Common class method), 359  word_iter() (translate.lang.de.de class method), 363  word_iter() (translate.lang.el.el class method), 364  word_iter() (translate.lang.es.es class method), 365  word_iter() (translate.lang.fa.fa class method), 366  word_iter() (translate.lang.fi.fi class method), 367  word_iter() (translate.lang.fr.fr class method), 367  word_iter() (translate.lang.gu.gu class method), 368  word_iter() (translate.lang.he.he class method), 369  word_iter() (translate.lang.hi.hi class method), 370  word_iter() (translate.lang.hy.hy class method), 371  word_iter() (translate.lang.km.km class method), 371  word_iter() (translate.lang.km.km class method), 372  word_iter() (translate.lang.kn.kn class method), 373  word_iter() (translate.lang.kn.kn class method), 374  word_iter() (translate.lang.ml.ml class method), 375  word_iter() (translate.lang.ml.ml class method), 375	words () (translate.lang.el.el class method), 364 words () (translate.lang.es.es class method), 365 words () (translate.lang.fa.fa class method), 366 words () (translate.lang.fi.fi class method), 367 words () (translate.lang.fr.fr class method), 367 words () (translate.lang.gu.gu class method), 368 words () (translate.lang.he.he class method), 369 words () (translate.lang.hi.hi class method), 370 words () (translate.lang.hy.hy class method), 371 words () (translate.lang.ja.ja class method), 371 words () (translate.lang.km.km class method), 372 words () (translate.lang.kn.kn class method), 373 words () (translate.lang.ml.ml class method), 374 words () (translate.lang.ml.ml class method), 375 words () (translate.lang.me.ne class method), 375 words () (translate.lang.ne.ne class method), 376 words () (translate.lang.si.si class method), 379 words () (translate.lang.st.st class method), 379 words () (translate.lang.st.st class method), 379 words () (translate.lang.st.st class method), 380
method), 356  word_iter() (translate.lang.common.Common class method), 359  word_iter() (translate.lang.de.de class method), 363  word_iter() (translate.lang.el.el class method), 364  word_iter() (translate.lang.es.es class method), 365  word_iter() (translate.lang.fa.fa class method), 366  word_iter() (translate.lang.fi.fi class method), 367  word_iter() (translate.lang.fr.fr class method), 367  word_iter() (translate.lang.gu.gu class method), 368  word_iter() (translate.lang.he.he class method), 369  word_iter() (translate.lang.hi.hi class method), 370  word_iter() (translate.lang.hy.hy class method), 371  word_iter() (translate.lang.ja.ja class method), 371  word_iter() (translate.lang.km.km class method), 372  word_iter() (translate.lang.kn.kn class method), 373  word_iter() (translate.lang.kn.kn class method), 374  word_iter() (translate.lang.ml.ml class method), 375  word_iter() (translate.lang.mr.mr class method), 375  word_iter() (translate.lang.mr.mr class method), 375  word_iter() (translate.lang.me.ne class method), 376	words () (translate.lang.el.el class method), 364 words () (translate.lang.es.es class method), 365 words () (translate.lang.fa.fa class method), 366 words () (translate.lang.fi.fi class method), 367 words () (translate.lang.fr.fr class method), 367 words () (translate.lang.gu.gu class method), 368 words () (translate.lang.he.he class method), 369 words () (translate.lang.hi.hi class method), 370 words () (translate.lang.hy.hy class method), 371 words () (translate.lang.ja.ja class method), 371 words () (translate.lang.km.km class method), 372 words () (translate.lang.kn.kn class method), 373 words () (translate.lang.ml.ml class method), 374 words () (translate.lang.ml.ml class method), 375 words () (translate.lang.mr.mr class method), 376 words () (translate.lang.ne.ne class method), 376 words () (translate.lang.si.si class method), 379 words () (translate.lang.si.si class method), 379 words () (translate.lang.si.si class method), 380 words () (translate.lang.sv.sv class method), 380 words () (translate.lang.ta.ta class method), 381
method), 356  word_iter() (translate.lang.common.Common class method), 359  word_iter() (translate.lang.de.de class method), 363  word_iter() (translate.lang.el.el class method), 364  word_iter() (translate.lang.fe.fe class method), 365  word_iter() (translate.lang.fi.fi class method), 366  word_iter() (translate.lang.fi.fi class method), 367  word_iter() (translate.lang.fr.fr class method), 367  word_iter() (translate.lang.gu.gu class method), 368  word_iter() (translate.lang.he.he class method), 369  word_iter() (translate.lang.hy.hy class method), 370  word_iter() (translate.lang.hy.hy class method), 371  word_iter() (translate.lang.km.km class method), 371  word_iter() (translate.lang.km.kn class method), 372  word_iter() (translate.lang.kn.kn class method), 373  word_iter() (translate.lang.ml.ml class method), 374  word_iter() (translate.lang.ml.ml class method), 375  word_iter() (translate.lang.mn.en class method), 376  word_iter() (translate.lang.ne.ne class method), 376  word_iter() (translate.lang.ne.ne class method), 376  word_iter() (translate.lang.ne.ne class method), 376	words () (translate.lang.el.el class method), 364 words () (translate.lang.es.es class method), 365 words () (translate.lang.fa.fa class method), 366 words () (translate.lang.fi.fi class method), 367 words () (translate.lang.fr.fr class method), 367 words () (translate.lang.gu.gu class method), 368 words () (translate.lang.he.he class method), 369 words () (translate.lang.hi.hi class method), 370 words () (translate.lang.hy.hy class method), 371 words () (translate.lang.ja.ja class method), 371 words () (translate.lang.km.km class method), 372 words () (translate.lang.kn.kn class method), 373 words () (translate.lang.ko.ko class method), 374 words () (translate.lang.m.mr class method), 375 words () (translate.lang.m.mr class method), 376 words () (translate.lang.ne.ne class method), 376 words () (translate.lang.si.si class method), 379 words () (translate.lang.st.st class method), 379 words () (translate.lang.st.st class method), 380 words () (translate.lang.ta.ta class method), 381 words () (translate.lang.ta.ta class method), 382
method), 356  word_iter() (translate.lang.common.Common class method), 359  word_iter() (translate.lang.de.de class method), 363  word_iter() (translate.lang.el.el class method), 364  word_iter() (translate.lang.es.es class method), 365  word_iter() (translate.lang.fa.fa class method), 366  word_iter() (translate.lang.fi.fi class method), 367  word_iter() (translate.lang.fr.fr class method), 367  word_iter() (translate.lang.gu.gu class method), 368  word_iter() (translate.lang.he.he class method), 369  word_iter() (translate.lang.hi.hi class method), 370  word_iter() (translate.lang.hy.hy class method), 371  word_iter() (translate.lang.ja.ja class method), 371  word_iter() (translate.lang.km.km class method), 372  word_iter() (translate.lang.kn.kn class method), 373  word_iter() (translate.lang.kn.kn class method), 374  word_iter() (translate.lang.ml.ml class method), 375  word_iter() (translate.lang.mr.mr class method), 375  word_iter() (translate.lang.mr.mr class method), 375  word_iter() (translate.lang.me.ne class method), 376	words () (translate.lang.el.el class method), 364 words () (translate.lang.es.es class method), 365 words () (translate.lang.fa.fa class method), 366 words () (translate.lang.fi.fi class method), 367 words () (translate.lang.fr.fr class method), 367 words () (translate.lang.gu.gu class method), 368 words () (translate.lang.he.he class method), 369 words () (translate.lang.hi.hi class method), 370 words () (translate.lang.hy.hy class method), 371 words () (translate.lang.ja.ja class method), 371 words () (translate.lang.km.km class method), 372 words () (translate.lang.kn.kn class method), 373 words () (translate.lang.ml.ml class method), 374 words () (translate.lang.ml.ml class method), 375 words () (translate.lang.mr.mr class method), 376 words () (translate.lang.ne.ne class method), 376 words () (translate.lang.si.si class method), 379 words () (translate.lang.si.si class method), 379 words () (translate.lang.si.si class method), 380 words () (translate.lang.sv.sv class method), 380 words () (translate.lang.ta.ta class method), 381

```
words () (translate.lang.ur.ur class method), 384
                                                        xmltags()
                                                                        (translate.filters.checks.MozillaChecker
words () (translate.lang.vi.vi class method), 385
                                                                 method), 321
words () (translate.lang.zh.zh class method), 386
                                                       xmltags()(translate.filters.checks.OpenOfficeChecker
wordsinunit () (in module translate.storage.statsdb),
                                                                 method), 326
         612
                                                       xmltags()
                                                                       (translate.filters.checks.ReducedChecker
WorkflowError, 672
                                                                 method), 332
wrap production()
                                                                       (translate.filters.checks.StandardChecker
                            (in
                                    module
                                                trans-
                                                       xmltags()
         late.storage.php), 552
                                                                 method), 338
wrapmessage()
                        (translate.convert.po2txt.po2txt
                                                       xmltags()
                                                                           (translate.filters.checks.TermChecker
         method), 258
                                                                 method), 345
write_odf() (in module translate.convert.xliff2odf),
                                                       XPathBreadcrumb
                                                                                   (class
                                                                                               in
                                                                                                        trans-
                                                                 late.storage.xml_extract.xpath_breadcrumb),
         267
                        (translate.tools.pydiff.DirDiffer
writediff()
         method), 698
                       (translate.tools.pydiff.FileDiffer
writediff()
         method), 698
                                                       yaml2po (class in translate.convert.yaml2po), 268
writexml_helper()
                            (in
                                    module
                                                trans-
                                                        Ζ
         late.misc.ourdom), 394
                                                        zfill()
                                                                          (translate.misc.multistring.multistring
X
                                                                 method), 390
X (class in translate.storage.placeables.base), 508
                                                        zh (class in translate.lang.zh), 385
X (class in translate.storage.placeables.xliff), 533
                                                        ZIPFile (class in translate.storage.zip), 681
XapianDatabase
                          (class
                                       in
                                                trans-
         late.search.indexing.XapianIndexer), 402
                         (class
XapianEnquire
                                      in
                                                trans-
         late.search.indexing.XapianIndexer), 404
xlifffile (class in translate.storage.xliff), 672
xliffunit (class in translate.storage.xliff), 675
xml_preserve_ancestors (in module
                                               trans-
         late.misc.xml helpers), 396
xml_space_ancestors
                              (in
                                     module
                                                trans-
         late.misc.xml helpers), 396
XMLEntityPlaceable
                              (class
                                        in
                                                trans-
         late.storage.placeables.general), 513
XmlNamer (class in translate.storage.xml_name), 681
XMLTagPlaceable
                           (class
         late.storage.placeables.general), 515
xmltags() (translate.filters.checks.CCLicenseChecker
         method), 274
xmltags()
                 (translate.filters.checks.DrupalChecker
         method), 280
                 (translate.filters.checks.GnomeChecker
xmltags()
         method), 286
                    (translate.filters.checks.IOSChecker
xmltags()
         method), 292
                    (translate.filters.checks.KdeChecker
xmltags()
         method), 298
                  (translate.filters.checks.L20nChecker
xmltags()
         method), 303
xmltags() (translate.filters.checks.LibreOfficeChecker
         method), 309
xmltags()
               (translate.filters.checks.MinimalChecker
         method), 315
```